UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Telecomunicaciones y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Análisis de tráfico de Internet mediante el uso del suavizado exponencial de series temporales

Eduardo Cabornero Pinto
Tutor: Luis de Pedro Sánchez
Ponente : Jorge López de Vergara

Julio 2021

Análisis del tráfico de Internet mediante el uso del suavizado exponencial de series temporales

AUTOR: Eduardo Cabornero Pinto
TUTOR: Luis de Pedro Sánchez

Computación y redes de altas prestaciones
Dpto. Tecnología Electrónica y de las comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio de 2021

Resumen (castellano)

La predicción del tráfico inalámbrico desempeña un papel clave en la gestión y planificación de redes, especialmente para la toma de decisiones en tiempo real y la predicción a corto plazo. Los sistemas requieren métodos de predicción de bajo coste, alta precisión y baja complejidad computacional. Además, el gran desarrollo de la tecnología y el tráfico en Internet han ido aumentando de manera exponencial y con ello los ataques en la red o ciberataques a lo largo este último siglo. Esto ha producido la necesidad del desarrollo de nuevas técnicas para detectar dichos ataques y proteger a nuestros dispositivos frente a ellos.

Precisamente este trabajo tiene como objetivo analizar el tráfico de Internet mediante el uso del suavizado exponencial de series temporales con el objetivo final y principal de desarrollar un algoritmo denominado Holt-Winters, también conocido como suavizado exponencial triple que es uno de los muchos métodos o algoritmos que se pueden utilizar para pronosticar puntos de datos en una serie temporal, siempre que la serie sea "estacional", es decir, repetitiva durante un período. El algoritmo de Holt-Winters destaca frente a otros métodos de pronóstico porque nos proporciona buenos resultados frente a todo tipo de tráficos y su uso es muy recomendable en la predicción del tráfico de la red celular. En este trabajo se trata de desarrollar y aplicar el algoritmo Holt-Winters a una serie temporal real y comprobar que la predicción del tráfico es correcta y que en caso de realizar un ataque dicho ataque se detecta correctamente. Además, se analizan otros métodos de pronóstico más simples y el suavizado exponencial simple y doble.

Este trabajo utiliza una trama de tráfico real y toma de partida el trabajo, los resultados y las conclusiones obtenidas por los autores previos. En dichos trabajos se trataba de conocer mejor el tráfico de red analizándolo mediante la distribución estadística alfa-estable.

Palabras clave (castellano)

Suavizado exponencial, Holt-Winters, MATLAB, Python, ventana, algoritmo, período.

Abstract (English)

Wireless traffic prediction plays a key role in network management and planning, especially for real-time decision making and short-term forecasting. Systems require prediction methods with low cost, high accuracy and low computational complexity. In addition, the great development of technology and Internet traffic has been increasing exponentially and with it the number of network attacks or cyber-attacks over the last century. This has produced the need for the development of new techniques to detect such attacks and protect our devices against them.

Precisely this work aims to analyze Internet traffic by using exponential smoothing of time series with the ultimate and main objective of developing an algorithm called Holt-Winters, also known as triple exponential smoothing, which is one of the many methods or algorithms that can be used to forecast data points in a time series, provided that the series is "seasonal", i.e. repetitive over a period of time. The Holt-Winters algorithm stands out from other forecasting methods because it provides good results for all types of traffic and its use is highly recommended for predicting cellular network traffic. In this work we try to develop and apply the Holt-Winters algorithm to a time series (in this case the data of the second week of June) and check that the traffic prediction is correct and that in case of an attack it is correctly detected. In addition, other simpler forecasting methods and single and double exponential smoothing are analyzed.

This work uses a real traffic frame and takes as a starting point the work, results and conclusions obtained by previous authors. In those works, the aim was to gain a better understanding of network traffic by analyzing it using the alpha-stable statistical distribution.

Keywords (inglés)

Exponential Smoothing, Holt-Winters, MATLAB, Python, window, algorithm, period.

Agradecimientos

Antes de empezar este trabajo me gustaría agradecer a mi tutor Luis de Pedro Sánchez por todo el apoyo mostrado durante el trabajo, especialmente en la situación actual

Por otro lado, quiero agradecer a mis amigos y a mi familia, en especial a mis padres y mi hermano por apoyarme en los momentos difíciles y por motivarme a seguir tratando de mejorar. Esta carrera me ha ayudado a valorar aún más a la gente que quiero.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

En esta sección se expondrá cual ha sido la motivación de hacer este TFG, los objetivos, las fases en las que ha sido realizado y la estructura del documento.

1.1 Motivación

La predicción del tráfico inalámbrico es un componente esencial de la planificación, el desarrollo y la gestión de redes. Una predicción precisa es aún más necesaria con el desarrollo de los sistemas inalámbricos de quinta generación(5G). Además, el desarrollo del sector tecnológico e informático en los últimos años ha venido acompañado de un creciente aumento de ciberataques, que cada vez aumentan más su alcance y su capacidad de pasar desapercibidos, afectando así a nuevos dispositivos que salen en el mercado y poniendo en peligro la infraestructura computacional de la red, los usuarios y su información. Se deben garantizar ciertos derechos como el derecho a la privacidad y el derecho a la integridad de la información para que el funcionamiento de la red sea correcto[1].

Un dato a destacar [2] con respecto a este crecimiento es que durante el tercer trimestre de 2020 se han detectado 3,3 millones de ataques de red según los servidores de Watchguard Technologies. Esto representa un crecimiento de 90% con respecto al trimestre anterior y el nivel más alto en dos años. Además, las firmas de ataques de red únicos también continuaron una trayectoria ascendente, alcanzando también el máximo de los dos últimos años. Así lo revela el Informe de Seguridad de Internet del tercer trimestre del año que acaba de presentar la compañía. En él, informa, además, de cómo la COVID-19 ha impactado en el panorama de amenazas de seguridad en Internet. Asimismo, deja constancia de que los atacantes continúan apuntando a las redes corporativas aumentando los dominios maliciosos relacionados con la pandemia. Todo lo anterior ha provocado que las organizaciones se hayan centrado en el mantenimiento y el fortalecimiento de las protecciones para los activos y servicios basados en la red mediante el desarrollo de algoritmos como en este caso el algoritmo de Holt-Winters.

1.2 Objetivos

El objetivo de este trabajo es analizar los datos y resultados en los trabajos de fin de grado anteriores para poder desarrollar el algoritmo de Holt-Winters y mediante este algoritmo poder realizar un pronóstico de la manera en la que se va a comportar una serie temporal real , comprobar que esa predicción es correcta y poder detectar un ataque en caso de que se produzca.

Este proceso se ha dividido en:

- Obtención de una serie temporal estacional.
- Obtención de un fichero de texto que incluya el tiempo, los bits por segundo y los paquetes por segundo.
- Desarrollo de diferentes métodos de pronóstico: método ingenuo, promedio simple, media móvil y media ponderada.
- Desarrollo suavizado exponencial simple.
- Desarrollo suavizado exponencial doble.
- Desarrollo suavizado exponencial triple o también llamado algoritmo de Holt-Winters.
- Analizar cómo se comporta el algoritmo de Holt-Winters frente a ataques.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1: Introducción:** En este capítulo se ha expuesto una breve introducción sobre la idea y los objetivos que tiene este TFG
- **Capítulo 2: Estado del arte:** Explica todas las ideas y conceptos necesarios para entender perfectamente el trabajo realizado
- **Capítulo 3: Diseño:** En este este capítulo se explicarán todas las herramientas que se han utilizado para la realización de este trabajo.
- **Capítulo 4: Desarrollo:** En este apartado se explicarán todas las etapas de desarrollo del trabajo y los motivos por los que se han realizado cada una de ellas.
- **Capítulo 5: Pruebas y resultados:** En esta sección se analizarán los resultados obtenidos.
- **Capítulo 6: Conclusiones y trabajo futuro:** a partir de los resultados obtenidos se habla de las conclusiones de este trabajo y de posibles trabajos futuros.

2 Estado del arte

2.1 Introducción

En este punto voy a presentar los conceptos más importantes para comprender el trabajo de realizado. Los puntos son:

- Métodos de pronóstico
- Suavizado exponencial
- Distancia
- Ciberataque

2.2 Métodos de pronóstico

En este punto voy a explicar unos métodos de pronóstico previos a la realización y el análisis del suavizado exponencial simple, el suavizado exponencial doble y el suavizado exponencial triple o algoritmo de Holt-Winters. Los diferentes métodos para pronosticar puntos son:

2.2.1 Método ingenuo

Se trata del método de pronóstico más primitivo y su premisa es que el punto esperado es igual al último punto observado:

$$\hat{y}_{x+1} = y_x$$

Este algoritmo se ha realizado el script de MATLAB método_ingenuo.m
Para representar estos métodos de pronóstico hemos utilizado la siguiente serie: serie= [3,10,12,13,12,10,12].
Por tanto, con este método el siguiente valor pronosticado es 12 y quedaría representado así:

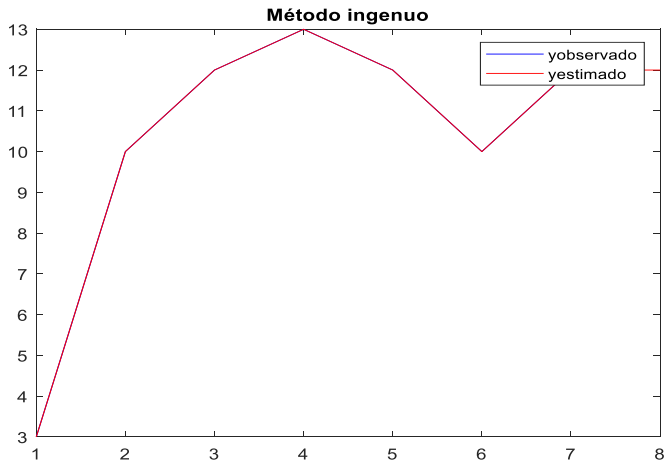

Figura 2-1: Demostración método ingenuo

2.2.2 Promedio Simple

Este método se basa en tomar los valores que tenemos, calcular el promedio, es decir, la media y pronosticar que el siguiente valor será igual al promedio calculado. La forma en la que se calcula el promedio es la siguiente:

$$\hat{y}_{x+1} = \frac{1}{x}\sum_{i=1}^{x} y_i$$

Este algoritmo se ha realizado el script de MATLAB Promedio_simple.m.
Para hacer una representación de este método se ha utilizado la serie utilizada en el método ingenuo para comprobar que el pronóstico es diferente. En este caso el valor pronosticado sería: 10.2857 y quedaría representado así:

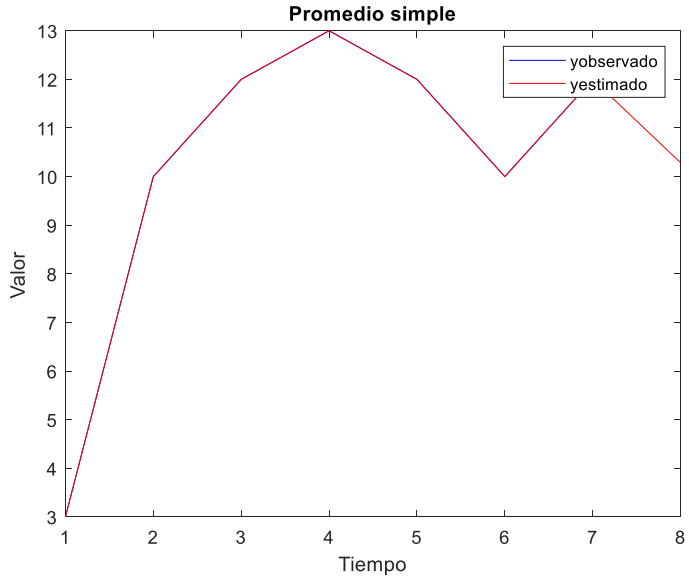

Figura 2-2: Demostración promedio simple

2.2.3 Media móvil

Es una mejora sobre el promedio simple y se basa en el promedio de los últimos n puntos, es decir, es un método en el que solo importan los valores recientes (los últimos n puntos). El cálculo de la media móvil implica el uso de lo que se denomina ventana móvil de n puntos. Este método puede ser muy útil si elige número n adecuado y es un método muy utilizado por los analistas de acciones.
Este algoritmo se ha realizado el script de MATLAB media_movil.m.
El resultado de aplicar este método a la misma serie utilizada en los métodos anteriores es que el punto pronosticado es: 11.3333 y quedaría representado de la siguiente forma:

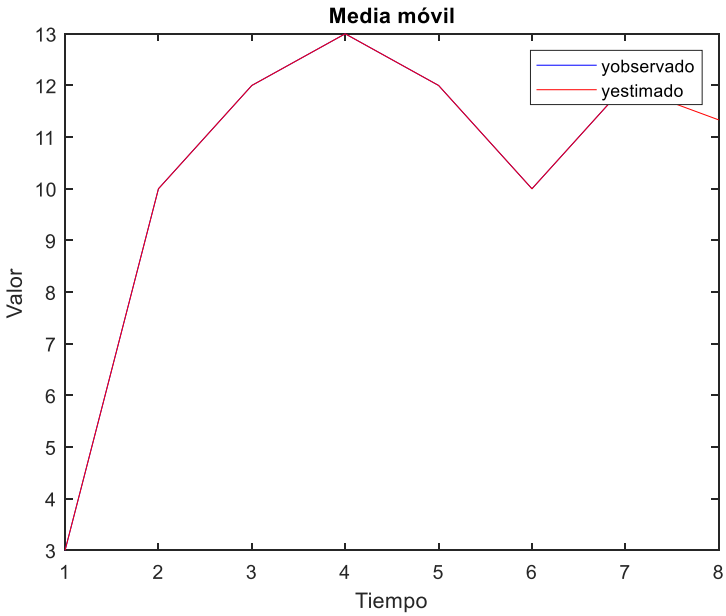

Figura 2-3: Demostración media móvil

2.2.4 Media móvil ponderada

La media móvil ponderada es una media móvil en la que, dentro de la ventana deslizante, los valores reciben diferentes ponderaciones, normalmente para que los puntos más recientes sean más importantes.

En este método en lugar de seleccionar un tamaño de ventana, requiere una lista de pesos (que deben sumar 1).

Este algoritmo se ha realizado en el script de MATLAB Media_movil_ponderada.m.

Por ejemplo, en el caso de seleccionar [0.1, 0.2, 0.3, 0.4] como pesos, lo que se está haciendo es dar un 10%, 20%, 30% y 40% a los últimos 4 puntos respectivamente. Utilizando la misma serie que en los métodos anteriores el pronóstico sería 11.5 y quedaría representado de la siguiente forma:

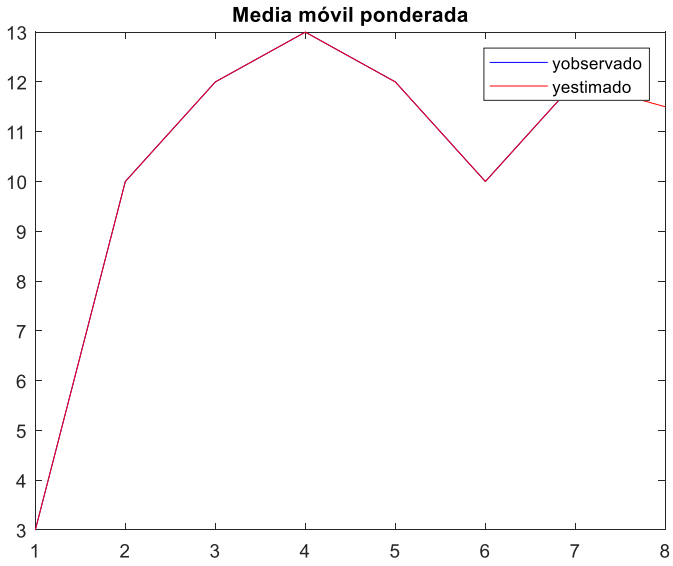

Figura 2-4 Demostración media móvil ponderada

2.3 Suavizado exponencial

El suavizado exponencial [3] se utiliza como un método para predecir un comportamiento futuro de una serie temporal a partir de los promedios históricos de una variable en un período, es decir, lo que se hace es suavizar dicha serie temporal con el objetivo de reducir las fluctuaciones y poder ver una tendencia que a simple vista no es clara. El suavizado exponencial es muy utilizado es el sector ventas ya que permite realizar previsiones de ventas con una eficacia muy notable.
Los métodos que utilizan suavizado exponencial destacan frente a otros mecanismos para la predicción del comportamiento de una serie temporal por su sencillez y facilidad de aplicación, aunque tiene otras múltiples ventajas:

- No es necesario tener muchos datos históricos, frente a otros métodos como por ejemplo ARIMA.
- Tiene más precisión frente a otros métodos debido a que utiliza técnicas de modelo exponencial.
- Tiene una gran flexibilidad ya que puede utilizar datos de demanda que pueden ser escogidos por el investigador.

Dentro del suavizado exponencial están el suavizado simple, doble y triple, pero antes para poder entender cada método hay que explicar algunos conceptos y su terminología:

- Serie: Es una secuencia ordenada de números.
- Valor observado vs Valor esperado: El suavizado exponencial se utiliza para estimar los valores que prevemos a partir de una serie dada. Los valores de la serie, es decir, los valores que conocemos son los valores observados y los valores esperados son los valores que pronosticamos a partir de esa serie dada.

2.3.1 Suavizado exponencial simple

El suavizado exponencial simple es una aplicación de la media ponderada móvil en la que asignamos pesos exponencialmente más pequeños a medida que aumentamos en el tiempo. En el suavizado exponencial simple se utiliza α (alpha) como si fuera el peso inicial en el método de la media móvil ponderada. α se llama coeficiente de suavizado o factor de suavizado.
La fórmula del suavizado exponencial simple es la siguiente:

$$\hat{y}_x = \alpha * y_x + (1 - \alpha) * \hat{y}_{x-1}$$

Donde: $\hat{y}_X$=valor esperado.
y_x= valor observado.

Por lo cual viendo la nomenclatura lo que tenemos es una media móvil ponderada con dos pesos α y (1- α). La suma de estos dos pesos es 1 (al igual que la media móvil ponderada) por lo que el método es correcto.
Una forma de entender el significado de α es analizar la propia expresión matemática. Cuando vemos la expresión anterior vemos que α es un valor que nos dice cuánto peso le damos al valor observado más reciente frente al último esperado, por tanto, se podría decir que α actúa como una palanca que da más peso al lado izquierdo cuando tiene un valor más alto (más cerca de 1) o al lado derecho cuando tiene un valor menor (más cerca de 0). Por tanto, analizando lo descrito anteriormente α se podría considerar como una tasa de deterioro de memoria, es decir, a mayor valor de α, más rápido "olvida" el método y dependiendo del valor de α tendremos una predicción u otra. El algoritmo fue programado en MATLAB en un script denominado prueba_exp_simple.m . A continuación, voy a aplicar el algoritmo de suavizado exponencial simple utilizando la serie utilizada en los métodos de pronóstico anteriores:

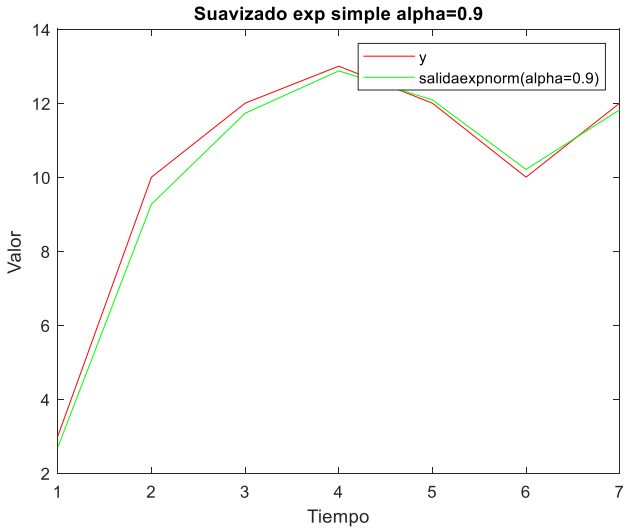

Figura 2-5: Suavizado exponencial simple (alpha=0.9)

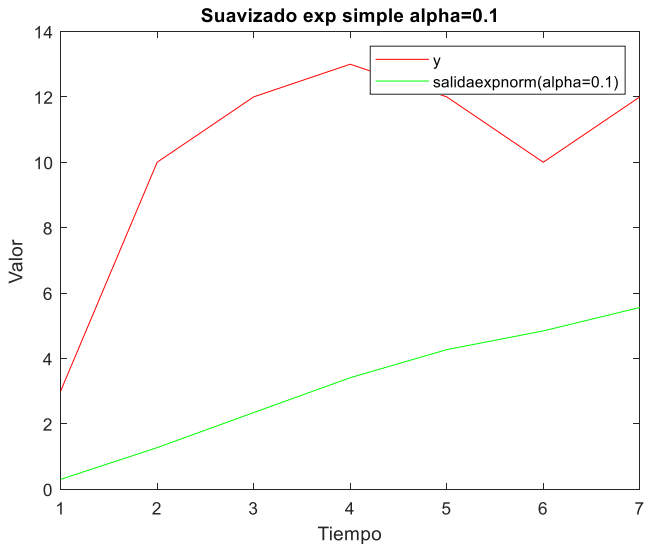

Figura 2-6: Suavizado exponencial simple(alpha=0.1)

Como vemos cuando utilizamos un valor de α más alto nuestra predicción (gráfica verde) estará más cerca de los valores observados (gráfica en rojo). Dependiendo de cada serie nos interesará un valor de α u otro, pero eso se tratará más adelante.

2.3.2 Suavizado exponencial doble

La diferencia principal con el suavizado exponencial simple es que nos permite predecir más de un punto, pero para ello hay que conocer antes algunos conceptos y su terminología:

- Nivel: En el suavizado exponencial doble el nivel es lo mismo que lo que es el valor esperado en el suavizado exponencial simple, pero como ahora solo es una parte del cálculo (ya que no predecimos un único punto como ocurría en el suavizado exponencial simple) lo denominaremos: l
- Tendencia o pendiente: En este algoritmo nos vamos a centrar en analizar puntos adyacentes por tanto la pendiente (b) se definiría de la siguiente forma:

  $b = y_x - y_{x-1}$

  donde y_x al igual que en el suavizado exponencial simple son los valores observados. Un factor que comentar es que a la hora de realizar el cálculo de la tendencia se puede hacer de dos formas: La primera sería el método multiplicativo en que en lugar de haber realizado la resta entre y_x- y_{x-1} hubiéramos hecho la división, pero a la hora de realizar el cálculo de la tendencia hemos decidido utilizar el método aditivo por simplicidad.

Ahora ya conocidos los dos términos anteriores ya podemos decir que el suavizado exponencial doble es un suavizado exponencial simple aplicado tanto a la tendencia como a el nivel. Esto que hemos descrito anteriormente se expresa de la siguiente forma:

$$l_x = \alpha * y_x + (1-\alpha) * (l_{x-1} + b_{x-1}) \quad Nivel$$
$$b_x = \beta * (l_x - l_{x-1}) + (1-\beta) * b_{x-1} \quad Tendencia$$
$$\hat{y}_{x+1} = l_x + b_x \quad Pronóstico$$

Como vemos la primera ecuación es la misma que la del suavizado simple con la diferencia de que ahora lo que sacamos es l en lugar de ŷ ya que ahora el valor esperado es la suma del nivel y la tendencia, mientras que en el suavizado simple no teníamos en cuenta la tendencia al solo tener que predecir un único punto.
En la segunda ecuación se usa una variable β que es el factor de tendencia. Dependiendo de los valores de α y β la predicción será mejor o peor (eso lo veremos más adelante).

Al igual que en el suavizado exponencial simple usábamos el primer valor observado como el primer valor esperado, en el suavizado exponencial doble utilizamos la primera tendencia observada como si fuera la primera tendencia esperada. Por lo cual es necesario tener mínimo dos puntos para poder calcular la tendencia inicial.
En el suavizado exponencial doble como tenemos un nivel y una tendencia podemos pronosticar dos puntos de datos y no uno (como ocurría en el suavizado exponencial simple).
El algoritmo fue programado en MATLAB en un script denominado SUAV_DOBLE.m . A continuación, vamos a hacer una representación de la predicción de la serie utilizada anteriormente utilizando suavizado doble para diferentes valores de α (factor de suavizado) y β (factor de tendencia):

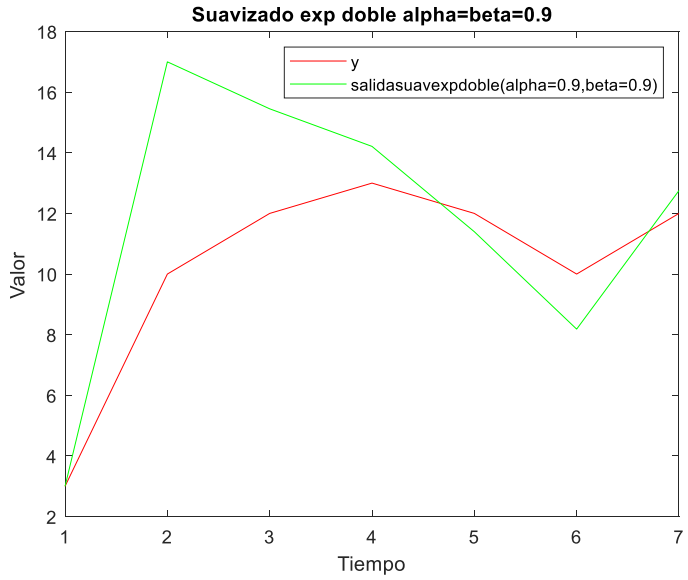

Figura 2-7: Suavizado exponencial doble(alpha=Beta=0.9)

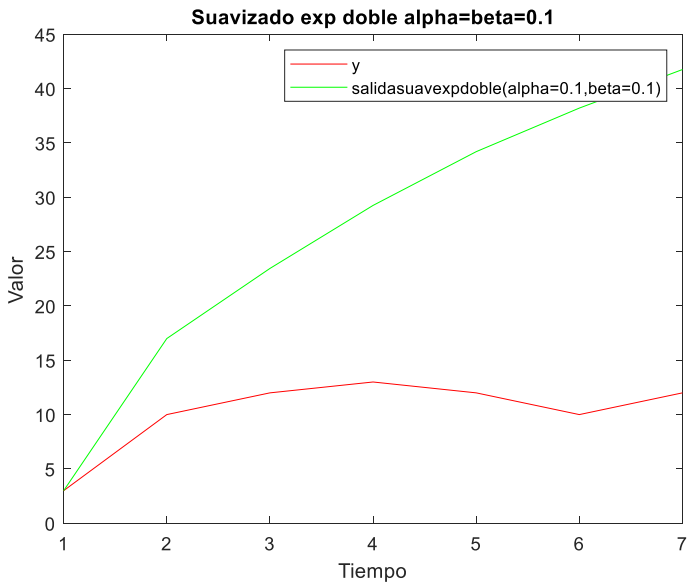

Figura 2-8 Suavizado exponencial doble (alpha, beta=0.1)

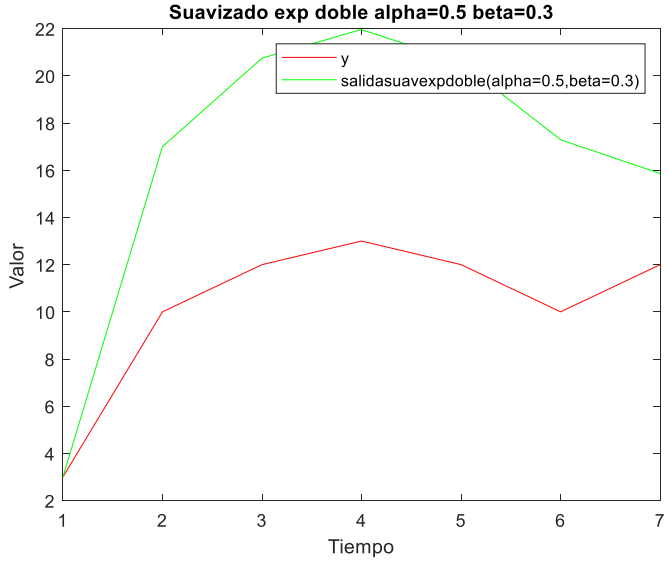

Figura 2-9: Suavizado exponencial doble (alpha=0.5, beta=0.3)

2.3.3 Suavizado exponencial triple

El suavizado exponencial triple o algoritmo de Holt-Winters consiste en aplicar el suavizado a las componentes estacionales además del nivel y la tendencia. Antes de entrar en las ecuaciones de este algoritmo hay que explicar algunos conceptos muy importantes.

- Temporada: Cuando una serie parece que se repite a intervalos regulares dicho intervalo se llama temporada. Para poder aplicar el algoritmo de Holt-Winters es necesario que dicha serie sea estacional. La duración de una temporada es la cantidad de puntos de datos después de los cuales comienza una nueva temporada. La duración de la temporada se hará con la siguiente nomenclatura: L

- Componente estacional: El componente estacional es una desviación adicional obtenida de sumar el nivel y la tendencia que se va repitiendo en el mismo desplazamiento en la temporada.
  Hay una componente estacional para cada punto de una temporada, es decir, si la duración de la temporada es 15, hay 15 componentes estacionales. La componente estacional se denotará con la siguiente nomenclatura: s.
  Un aspecto que hay que tener muy cuenta a la hora de aplicar el algoritmo de Holt-Winters es que si por ejemplo queremos predecir el tercer punto de la temporada 46 de una serie no podemos utilizar la componente estacional del tercer punto de la temporada 45, en el caso de que ese tercer punto haya sido pronosticado, es decir, solo podemos utilizar el último conjunto de componentes estacionales de los puntos observados.

El suavizado exponencial triple se aplica entre temporadas, por ejemplo, el componente estacional del cuarto punto en la temporada se suavizaría exponencialmente con el cuarto punto de la temporada pasada, el cuarto punto de hace dos temporadas etc. Esto matemáticamente se representa de la siguiente forma:

$$l_x = \alpha * (y_x - s_{x-L}) + (1-\alpha) * (l_{x-1} + b_{x-1}) \qquad Nivel$$
$$b_x = \beta * (l_x - l_{x-1}) + (1-\beta) * b_{x-1} \qquad Tendencia$$
$$s_x = \gamma * (y_x - l_x) + (1-\gamma) * s_{x-L} \qquad Componente\ estacional$$
$$\hat{y}_{x+m} = l_x + m * b_x + s_{x-L+(m-1) mod L} \qquad Pronóstico$$

Viendo la expresión matemática hay nuevos conceptos con respecto al suavizado exponencial doble y al suavizado exponencial simple. El primero es la aparición de una nueva variable: γ (gamma) que es el factor de suavizado para la componente estacional. Otro aspecto nuevo es que en la cuarta ecuación (la ecuación de pronóstico) vemos que podemos predecir (X+m) puntos, mientras que en el suavizado exponencial simple solo podemos predecir un punto y en el doble solo dos puntos. Una vez explicado el algoritmo de Holt-Winters y sus ecuaciones se ha realizado una predicción utilizando una serie estacional como la siguiente:

series = [30,21,29,31,40,48,53,47,37,39,31,29,17,9,20,24,27,35,41,38,
27,31,27,26,21,13,21,18,33,35,40,36,22,24,21,20,17,14,17,19,
26,29,40,31,20,24,18,26,17,9,17,21,28,32,46,33,23,28,22,27,
18,8,17,21,31,34,44,38,31,30,26,32]

La representación de la serie es:

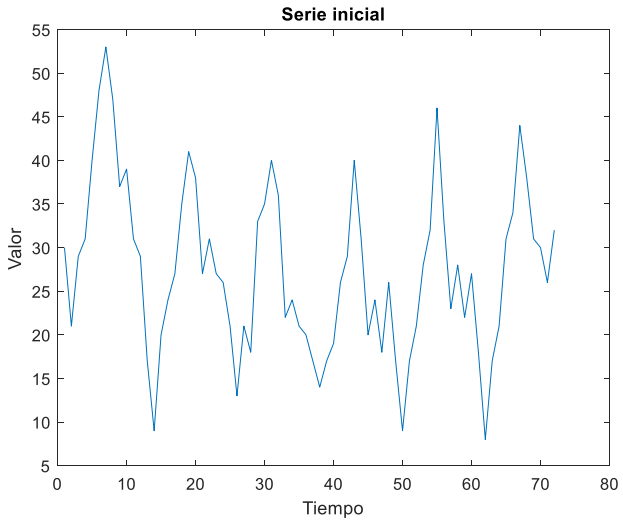

Figura 2-10: Serie Holt-Winters ejemplo

Como se aprecia en la serie la duración de la temporada, es decir, el período es 12. Una vez conocidas la duración de la temporada, las ecuaciones explicadas en el diseño que incluyen el nivel, la tendencia y la estacionalidad hay que obtener los valores de las variables matemáticas (α, β, γ) que nos dan una mejor predicción. En este ejemplo esos valores ya los conocemos (α =0.716, β = 0.029, γ = 0.993), pero depende totalmente de la serie. En este caso hemos tratado de predecir 24 puntos (2 períodos) y la predicción ha sido la siguiente:

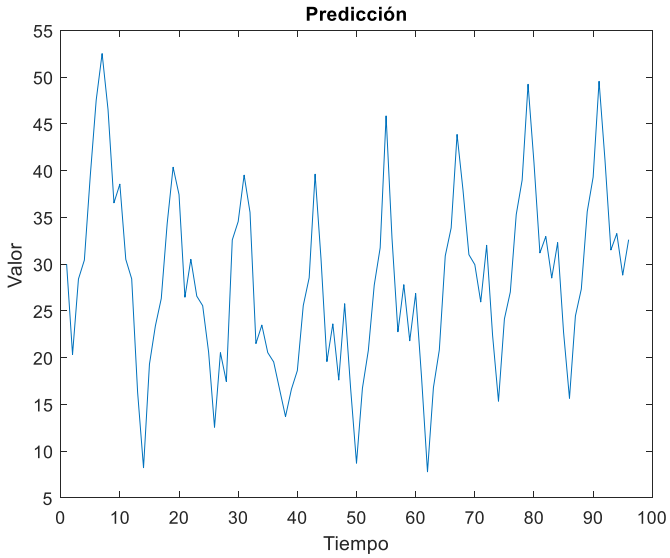

Figura 2-11: Predicción Holt-Winters ejemplo

Como se ve se han predicho 24 puntos a partir de los 72 puntos de la serie inicial y los resultados obtenidos son muy buenos ya que sigue la misma tendencia que las 6 temporadas anteriores (cada temporada tiene una duración de temporada igual a 12).

2.4 Distancia

La distancia [4] entre dos puntos del espacio euclídeo equivale a la longitud del segmento de la recta que los une expresado de manera numérica. A continuación, vamos a tratar el concepto de distancia en espacios euclídeos, aunque también se podría tratar en otros espacios más complejos, pero no es de interés a la hora de entender el trabajo realizado.
El término distancia es importante en nuestro trabajo porque se utiliza en el algoritmo de Holt-Winters para obtener los valores de las variables matemáticas α, β, γ para obtener una predicción adecuada. Cuando hemos calculado la predicción lo que hemos hecho es calcular la diferencia entre los puntos predichos y los valores reales esperados y utilizar la diferencia más grande en valor absoluto como la distancia. A continuación, lo explico:

Distancia en la recta
Existe una biyección, es decir, una correspondencia elemento a elemento entre los puntos de una recta y el conjunto de los números reales, de manera que a cada número real le corresponde un solo punto y a cada punto exactamente un número real. Para poder realizar esto es necesario un punto O y fijo de la recta y otro punto U, tal que por definición 1 es la abscisa de U. Se denota U(1). A la izquierda tenemos los puntos de abscisa negativos, a la derecha los positivos, mientras el origen O tiene abscisa 0. Dicha recta provista de abscisas para cada uno de sus puntos se conoce como recta real.
Si A(X_1) y B(X_2) son dos puntos de la recta real, la distancia entre A y B es la siguiente:
d(A,B)=| X_2- X_1|.

2.5 Ciberataques

Los ciberataques [5] o también conocidos como ataques informáticos son maniobras que son realizados a través de la red, cuyo objetivo es causar un daño en un sistema informático sin tener ningún tipo de autorización. Los ataques informáticos pueden ser de varios tipos según su objetivo. Estos pueden ser: tomar el control o desestabilizar un sistema informático para corromper o incluso eliminar archivos, datos privados o algoritmos de dicho sistema.
Algunos de ellos son el ataque de denegación de servicio [6] y su versión distribuida, los cuales se explicarán a continuación debido su gran relación con el trabajo realizado.

Ataque de denegación de servicio (DoS): Se trata de un tipo de ataque informático a una red cuya finalidad es hacer inaccesible un servicio o recurso a los usuarios que lo solicitan. Los servidores de la red solo pueden atender un número de peticiones simultáneamente y si se supera este límite, el sistema puede ralentizarse o incluso llegar a bloquearse o desconectarse de la red.

Ataque distribuido de denegación de servicio (en inglés *Distributed Denial of Service,* DDoS): Se trata de una evolución del ataque DoS que lo que hacen es emitir una gran cantidad de flujos de información a un mismo destino desde varios puntos al mismo tiempo. Este ataque es mucho más difícil de detectar y de bloquear que los ataques DoS, ya que este ataque informático proviene de diferentes direcciones IP, y además tiene una mayor capacidad de poder derribar el servicio de la red ya que al disponer de un amplio número de máquinas la cantidad de peticiones va a ser mucho mayor que la de un ataque DoS.

Estos ataques se suelen realizar mediante una red de “bots” que son los encargados de mandar flujos de datos enormes provocando que los servidores no sean capaces de atender todas las solicitudes entrantes y terminen colapsando[7]

3 Diseño

3.1 Introducción

Este capítulo tiene como objetivo explicar el diseño que se ha utilizado en cada una de las partes del trabajo, las diferentes posibilidades que existían a la hora de realizar cada una de esas partes y explicar los motivos que se han tenido en cuenta a la hora de tomar unas decisiones u otras.

El diseño de este trabajo se divide en los siguientes puntos:

- AWK
- PYTHON
- MATLAB
- Ventanas temporales

3.2 AWK

En primer lugar, a la hora de empezar a realizar el trabajo hubo que tomar una decisión acerca del software en el que se desarrollaría el trabajo. En un principio teniendo en cuenta los conocimientos previos y que todo el código de partida de los trabajos anteriores era en MATLAB decidimos empezar en MATLAB, pero una vez comenzado el trabajo se llegó a la conclusión de que los datos proporcionados por los trabajos anteriores (estos datos habían sido obtenidos gracias a la monitorización del tráfico de red de la Universidad de Granada durante 7 días) no nos iban a ser útiles para el objetivo final de este trabajo. El motivo por el que los datos proporcionados por los trabajos anteriores no nos eran de utilidad es que para poder aplicar el algoritmo de Holt-Winters la serie sobre la que se aplica este algoritmo debe ser estacional y esos datos no nos proporcionaban esa estacionalidad.

En un principio pensamos diferentes posibilidades para poder usar esos datos y así ahorrar trabajo, pero finalmente tomamos la decisión de escoger otros datos de la misma Universidad de Granada, pero de diferente semana y con el objetivo claro de que lo importante era obtener una serie estacional. Uno de los posibles motivos por el que creemos que dichas series utilizadas en los trabajos anteriores no son estacionales puede ser el simple hecho de que los datos utilizados se tomaron en una semana "rara". Cuando hablo de semana rara me puedo referir por ejemplo a una semana de Navidades, Semana Santa o una semana en la que haya un puente, ya que en estas semanas puede haber cambios en cuanto al tráfico de red ya sea por cierre de las empresas, por fiestas, horarios reducidos etc. Por lo tanto, la decisión que tomamos fue tomar los datos de la segunda semana de Junio [6] ya que esa semana no hay ningún evento, fiesta que pueda suponer un cambio en cuanto a lo que es el tráfico normal de Internet. Una vez descargado ese fichero (june.week2.csv) lo que se hico es ver en el dataset [8] los campos que nos interesan y hacer un fichero AWK denominado serietemporal.sh que nos proporcione como resultado un fichero de denominado serietemporalJunio.txt formado por tres columnas: la primera columna con el tiempo en segundos, la segunda con los bits por segundo(bits/segundo) y la tercera con los paquetes por segundo (paquetes/segundo).
Una vez que se obtuvo la predicción de los 10 y 15 primeros minutos de la tercera ventana de la segunda semana de Junio se pasó a introducir ataques para comprobar que el algoritmo de Holt-Winters puede detectarlos. Para ello lo que se hico es utilizar el mismo fichero AWK

utilizado anteriormente y aplicarlo a los ataques que hay en la misma página en la que habíamos obtenido los datos de la segunda semana de Junio.

3.3 MATLAB

Como he dicho anteriormente el código proporcionado por los trabajos anteriores estaba en MATLAB y eso junto al mayor dominio de MATLAB que de otros lenguajes habían sido los factores principales por los que la idea era solamente utilizar este software a la hora de programar. Finalmente, como he explicado anteriormente tuve que utilizar otros lenguajes para obtener el fichero serietemporalJunio.txt que nos da tres columnas: tiempo, bits/segundo y paquetes/segundo de los datos relativos a la segunda semana de Junio de la Universidad de Granada.

Antes de pasar directamente a desarrollar el algoritmo de Holt-Winters o también conocido como suavizado exponencial triple lo que se hizo es realizar el suavizado exponencial simple [9] y el suavizado exponencial doble [10] en MATLAB cuyas ecuaciones como he explicado anteriormente nos servirán para desarrollar el algoritmo de Holt Winters.

En los trabajos anteriores se había llegado a la conclusión de que las ventanas temporales de 5 minutos eran las ventanas temporales cuya duración permitía sacar los mejores parámetros α-estables, de manera que se pudiesen diferenciar perfectamente las zonas de tráfico normal y tráfico de ataque. En este trabajo al utilizar datos distintos y como el objetivo final era desarrollar el algoritmo de Holt-Winters, el cual nos exige una estacionalidad en la serie a analizar decidimos utilizar una ventana más larga: 15 minutos, y así poder ver si la estacionalidad de la serie se produce porque la serie es realmente estacional y no porque justo en la ventana temporal de 5 minutos de la casualidad de que la serie es estacional.
Una vez comprobado que utilizando ventanas temporales de 15 minutos la serie es estacional se decidió que esa ventana era correcta. Finalmente se utilizó una ventana de 30 minutos para hacer comprobaciones y ver si se obtenían buenos resultados.

A la hora de desarrollar el algoritmo de Holt Winters hay un aspecto muy importante a tener en cuenta como es la duración de la temporada, es decir, el período. Como en los datos proporcionados por la segunda semana de Junio no se podía apreciar exactamente la duración de la temporada fue necesario realizar dos algoritmos para calcular el período utilizando MATLAB: calculo_periodo.m y calculo_periodo_Segundo_Metodo.m. Los resultados obtenidos por estos dos métodos no eran del todo precisos ya que una de las variables de los algoritmos era el período esperado, pero igualmente nos sirvió para hacernos una idea sobre el valor de la duración de la temporada y a partir de ahí tratar de obtener la mejor predicción posible.

En un principio la idea fue ir prediciendo un número de puntos iguales al período de la serie y con series cuya duración sea una ventana completa ya que el período que habíamos calculado anteriormente con los dos algoritmos anteriores no era muy preciso. Una vez que se observó cual era el período de la serie se procedió a predecir más tiempo y una vez obtenida esa predicción se comprobó si se podía detectar un ataque.

3.4 PYTHON

En un principio como he explicado anteriormente la idea era desarrollar todo el código en MATLAB ya que todo el código proporcionado por los trabajos anteriores se desarrolla en este lenguaje, pero como hemos dicho anteriormente finalmente los datos utilizados anteriormente no nos servían para aplicar el algoritmo de Holt-Winters.
Finalmente se tomó la decisión de implementar el algoritmo de Holt-Winters en Python (aunque en primer lugar se desarrolló en Matlab) debido a que al utilizar un número mayor ecuaciones y al tener más variables matemáticas (α, β, γ) que el suavizado simple y doble era mucho más útil el uso de funciones para simplificar el código.

Otro aspecto a la hora de utilizar Python frente a MATLAB a la hora de obtener las variables matemáticas (α, β, γ) que varían entre 0 y 1 fue necesario ver cuáles eran los valores específicos para obtener la mejor predicción posible.

En principio lo que se hico fue predecir únicamente una temporada, es decir, un período a partir de una ventana completa. Con este procedimiento se pretendía ver cuál era decidir cuál era la duración de la temporada y el período que mejores resultados nos podría dar a la hora de obtener la predicción de los 10 primeros minutos de la tercera ventana de Junio. Una vez obtenida buena predicción de los 10 primeros minutos de la tercera ventana se predijo los 15 primeros minutos y se introdujo un ataque DoS de duración 10 minutos con la idea de tratar de detectarlo. Un aspecto importante es que en todas las simulaciones se ha utilizado como serie una ventana completa, en la predicción de los 10 y 15 primeros minutos de la tercera ventana de la segunda semana de Junio se ha tomado como serie inicial la segunda ventana.

3.5 Metodología

En este punto se va a explicar el proceso que se ha seguido para realizar el trabajo completo:

- En primer lugar, fue necesario obtener un fichero AWK con las características que nos interesaban para nuestros objetivos. Este fichero se denomina serietemporal.sh ya que los datos proporcionados por los trabajos anteriores no nos servían para nuestro objetivo final ya que no eran series estacionales.
- En segundo lugar, se desarrollaron el suavizado exponencial simple y el suavizado exponencial doble junto a otros métodos de pronóstico más simples utilizando MATLAB.
- En tercer lugar, se desarrollaron dos algoritmos. Estos algoritmos se desarrollan en los scripts de MATLAB:calculo_periodo.m y calculo_periodo_Segundo_Metodo.m. y se utilizan para poder conocer la duración de la temporada o también conocido como el período.
- Una vez ya conocido la duración de la temporada se procedió a realizar el algoritmo de Holt-Winters en PYTHON utilizando ventanas de 15 y 30 minutos con diferentes períodos hasta conseguir una predicción buena.
- Finalmente se realizó una predicción de 10 y 15 minutos y se introdujo un ataque DoS de 10 minutos y se comprobó que con este algoritmo se podía detectar.

3.6 Conclusiones

En este capítulo se han ido explicando todos las consideraciones y etapas necesarias para poder realizar el trabajo junto al software y el lenguaje de programación que se han utilizado. En los siguientes capítulos se explicará y analizará con detalle cada uno de los procedimientos y desarrollos que se han utilizado para conseguir el objetivo final de este trabajo.

4 Desarrollo y pruebas

4.1 Introducción

En este capítulo se explicará con detalle todos los procedimientos que se han realizado en este trabajo incluyendo funciones, ajustes realizados y análisis de resultados. Para ello se han tenido en cuenta las decisiones que se habían tomado anteriormente en la etapa de diseño. Los pasos que se han ido realizando en el desarrollo son los siguientes:

- Series temporales
- Suavizado exponencial triple o algoritmo de Holt-Winters

4.2 Series temporales

En esta primera parte del trabajo lo que se hizo fue obtener un fichero denominado serietemoralJunio.txt que incluyera tres columnas: La primera con el tiempo en segundos, la segunda con los bits por segundo y la tercera con los paquetes por segundo. Ésto se hizo ya que el objetivo final era poder aplicar el suavizado exponencial triple o algoritmo de Holt-Winters, y para ello era necesario tener una serie temporal estacional.
Como he comentado en el diseño los datos a partir de los cuales se partieron para obtener ese fichero .txt fueron los referentes al fichero june_week2_csv. Este fichero .csv es un fichero de texto en el que los diferentes campos de la segunda semana de Junio vienen separados por comas.
Una vez descargado este fichero .csv lo que se hizo es analizar cada uno de los campos que contenía cada línea de ese fichero, para ello se tuvo que hacer un estudio del dataset [7] en el que en cada línea destacan características como: tiempo del final del flujo(te), duración del flujo (td), dirección IP de origen (sa), dirección IP de destino (da), puerto de origen (sp), puerto de destino (dp), protocolo (pr), banderas (flg), estado de reenvío (fwd), tipo de servicio (stos), paquetes intercambiados en el flujo (pkt) y los bytes intercambiados en el flujo (byt).
Como lo que nos interesa es obtener un fichero de texto con los bits por segundo y los paquetes por segundo se ha tenido en cuenta únicamente las siguientes características del dataset: la duración del flujo (td), los bytes intercambiados en el flujo(byt) y los paquetes intercambiados en el flujo (pkt) y el tiempo final del flujo (te). Esto se hico por medio del fichero AWK serietemporal.sh,

A continuación, lo que hice fue obtener un fichero de menor tamaño denominado cienmillineas.txt que que incluyera las 100000 líneas del fichero .csv de la segunda semana de Junio para así poder analizar la estructura que tiene cada una de las líneas de este fichero y así poder calcular los bits por segundo y los paquetes por segundo. Un ejemplo de una línea de este fichero podría ser el siguiente (es la primera línea del fichero cienmillineas.txt):

2016-06-06
00:05:50,4.420,210.94.198.105,42.219.156.211,34471,443,TCP,.AP.S.,0,0,9,1564,background

La metodología que se ha usado para calcular los bits por segundo en este caso es la siguiente: En este caso la duración del flujo (td) es igual 4.420 segundos, los bytes intercambiados en el flujo(byt) son 1564 y el día y hora del tiempo final del flujo es el 6 de

Junio de 2016 a las 00:05:50, por tanto lo que hay que hacer es dividir los bytes por segundo(1564 bytes) entre la duración del flujo(4.420 segundos) y así se obtendrán los bytes por segundo de los últimos 4 segundos mientras que para calcular los bytes por segundo del primer segundo incompleto: 4.420-4=0.420 segundos se calculará dividiendo esa parte decimal de la duración del flujo(0.420 segundos) entre la duración del flujo(4.420 segundos) y multiplicando el resultado por los bytes por segundo.
Por lo tanto, en este caso los segundos 50, 49,48 y 47 de las 00:05 del 6 de Junio de 2016 tendrán 1564:4.420=353.846 bytes por segundo o lo que viene a ser 8*353.846 bits por segundo, mientras tanto el segundo 46 de las 00:05 del 6 de Junio de 2016 tendrán 1564*(0.420:4.420) =148.615 bytes por segundo o lo que viene a ser 8*148.645 bits por segundo.
Este mismo procedimiento se ha utilizado para calcular los paquetes por segundo y para cada uno de los segundos pertenecientes a la segunda semana de Junio.

Otros aspectos que se tuvieron en cuenta para realizar el fichero AWK fueron el uso de la función de Python split () que nos devuelve una lista y la función mktime () para conseguir obtener la fecha en segundos.
Una vez realizado el fichero serietemporal.sh que nos proporcionaba un fichero seritemporalJunio.txt se representó en MATLAB la primera ventana (teniendo en cuenta ventanas de 15 minutos) de la segunda semana de Junio obteniendo el siguiente resultado:

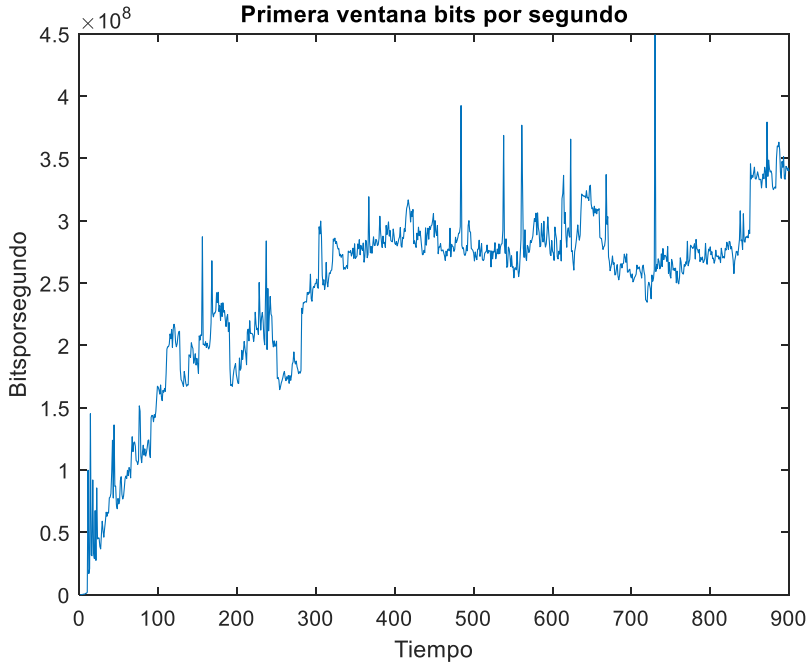

Figura 4-1: Primera ventana en bps

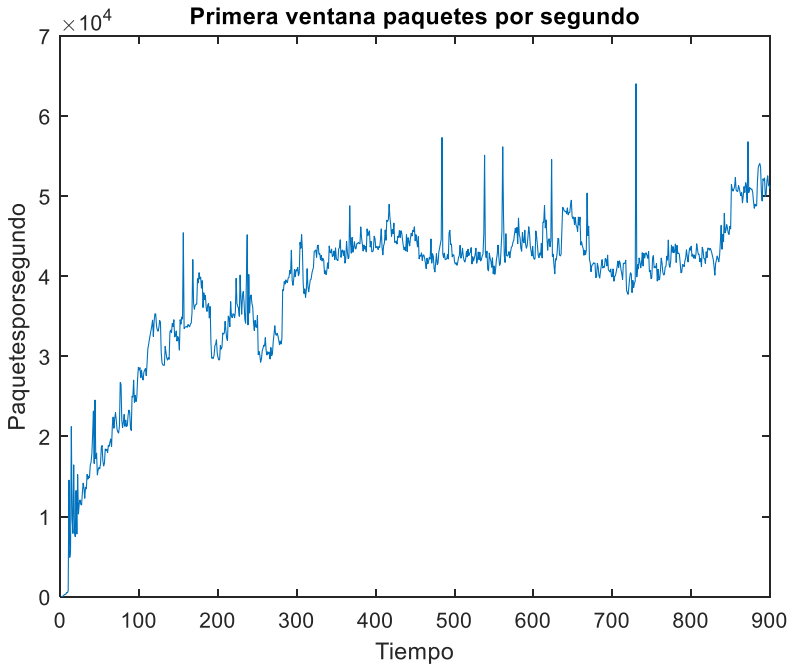

Figura 4-2: Primera ventana en paquetes/segundo

Como se ve en las gráficas en la primera ventana no se aprecia una estacionalidad muy clara (la cual es necesaria para poder aplicar más tarde el algoritmo de Holt-Winters). Por tanto, lo que se hico fue representar la segunda ventana obteniendo los siguientes resultados:

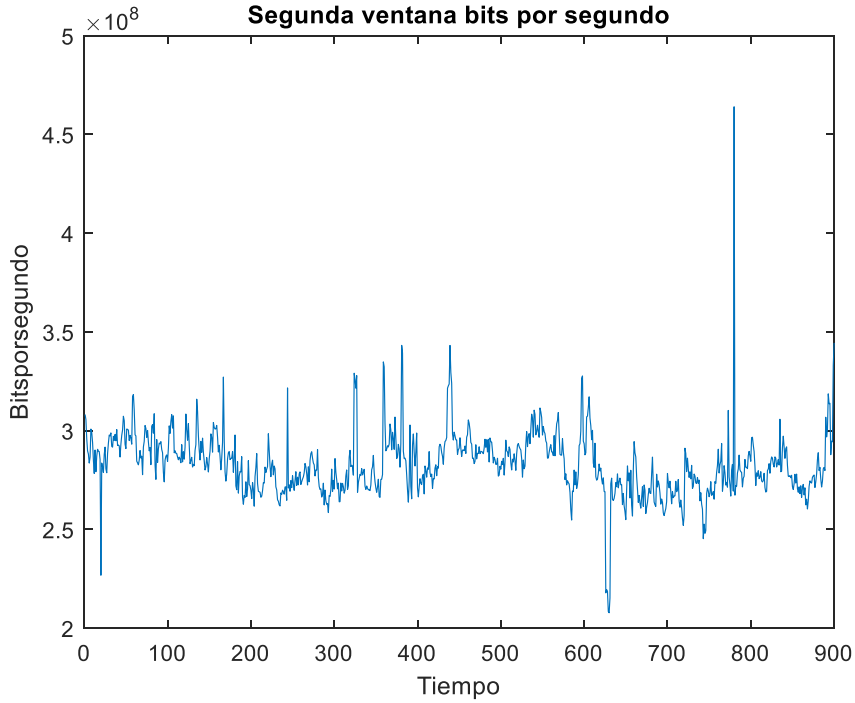

Figura 4-3: Segunda ventana en bps

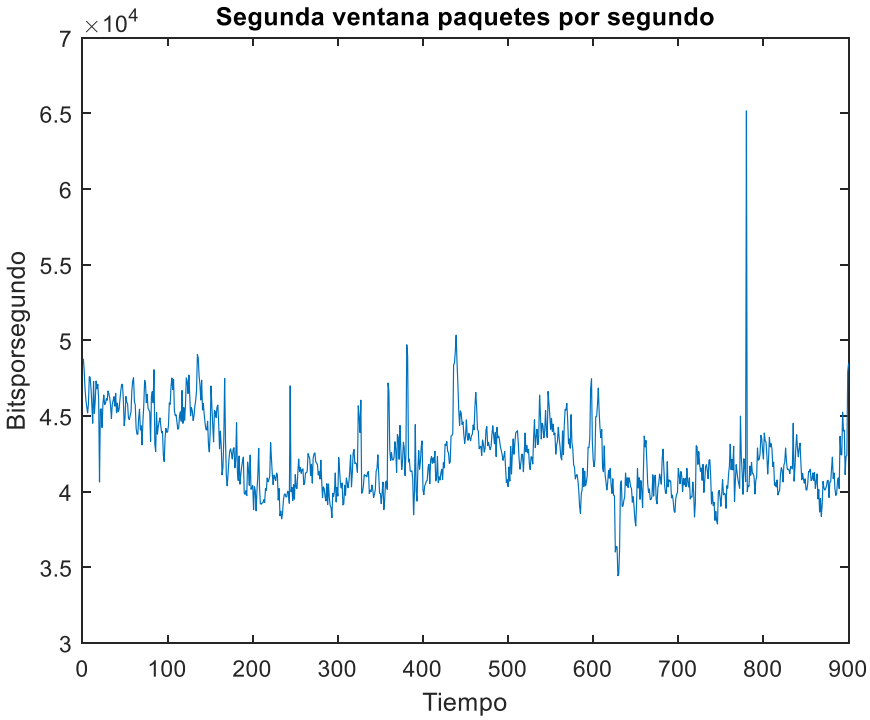

Figura 4-4: Segunda ventana en paquetes/segundo

Como se aprecia tanto en la gráfica de la segunda ventana de bits por segundo y de paquetes por segundo ya hay una estacionalidad, aunque no se aprecie el valor exacto a primera vista.

En los trabajos anteriores se habían utilizado ventanas de 5 minutos para así obtener los mejores parámetros α-estables, de manera que se pudiesen diferenciar perfectamente las zonas de tráfico normal, tráfico de ataque y tráfico mixto. En el capítulo del diseño ya expuse el motivo por el que en un principio se pensó que sería más útil el uso de ventanas de 15 minutos frente a las de 5, pero igualmente decidimos representarlo en MATLAB para comprobar ese aspecto:

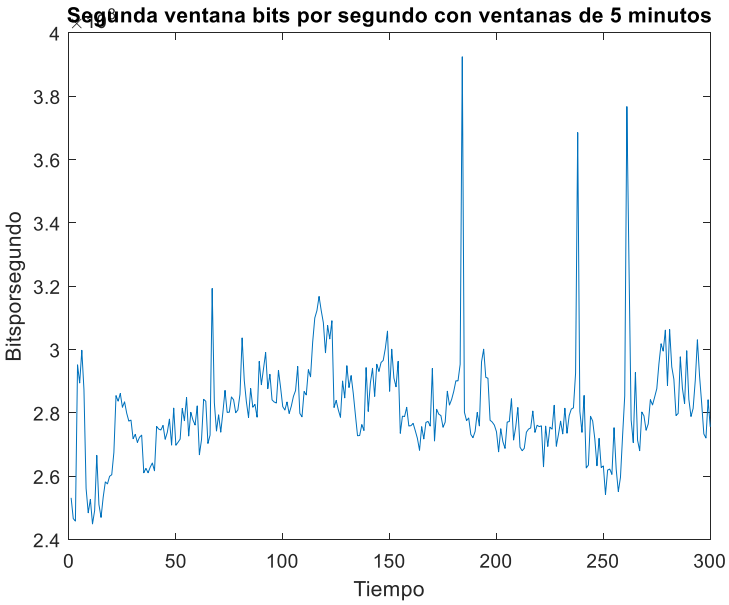

Figura 4-5: Segunda ventana de 5 minutos en bps

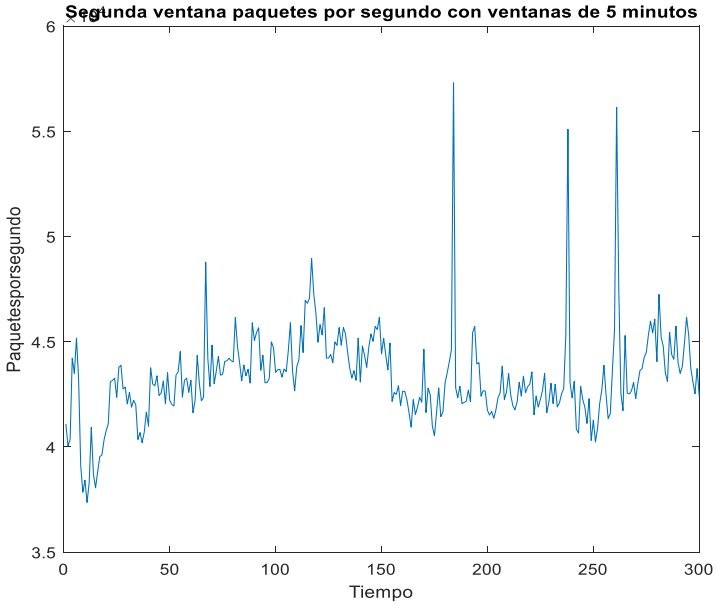

Figura 4-6: Segunda ventana de 5 minutos en paquetes/segundo

Como se aprecia hay una cierta estacionalidad, pero a simple vista se ve que es menos estacional que cuando se ha utilizado anteriormente ventanas de 15 minutos. Este factor junto a la posibilidad de que al utilizar ventanas más pequeñas tengamos menos precisión nos hico decantarnos por el uso de ventanas de 15 minutos.

Posteriormente se realizó un estudio sobre todas las ecuaciones y conceptos del suavizado exponencial simple, doble y los métodos de pronóstico más simples se pasó a analizar el suavizado exponencial triple o algoritmo de Holt-Winters el estado del arte que dependiendo.

La explicación de estos métodos se encuentra en el estado del arte.

4.3 Suavizado exponencial triple o algoritmo de Holt-Winters

Una vez ya analizado el suavizado exponencial simple, el suavizado exponencial doble y los diferentes métodos de pronóstico, se pasó al objetivo final de tratar de predecir más de un punto mediante el suavizado exponencial triple o el algoritmo de Holt-Winters. En el suavizado exponencial simple solo se podía predecir un punto y en el suavizado exponencial doble solo dos puntos. En el suavizado exponencial triple o algoritmo de Holt-Winters podemos predecir más de dos puntos.

En primer lugar, lo que se ha hecho es tratar de aplicar el algoritmo de Holt-Winters al tutorial que se ha seguido para comprobar que funciona. Como para poder aplicar este algoritmo hemos utilizado una serie estacional. La serie es la siguiente:

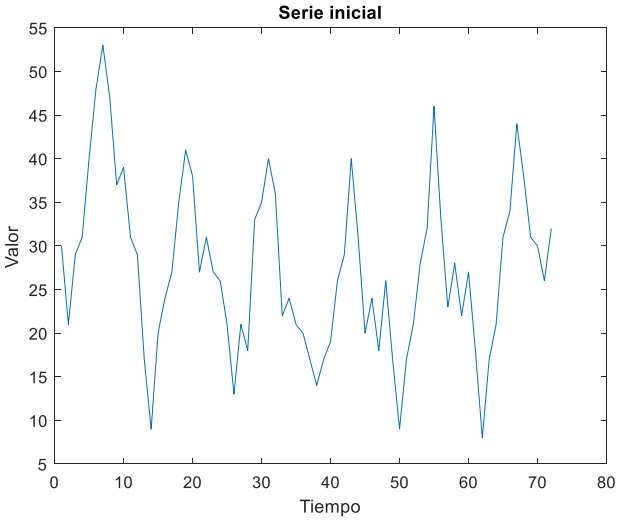

Figura 4-7: Serie estacional

Como se aprecia en la serie la duración de la temporada, es decir, el período es 12. Una vez conocidas la duración de la temporada, las ecuaciones explicadas en el diseño que incluyen el nivel, la tendencia y la estacionalidad hay que obtener los valores de las variables matemáticas (α, β, γ) que nos dan una mejor predicción. En este ejemplo esos valores ya los conocemos (α =0.716, β = 0.029, γ = 0.993), pero depende totalmente de la serie. En este caso hemos tratado de predecir 24 puntos (2 períodos) y la predicción ha sido la siguiente:

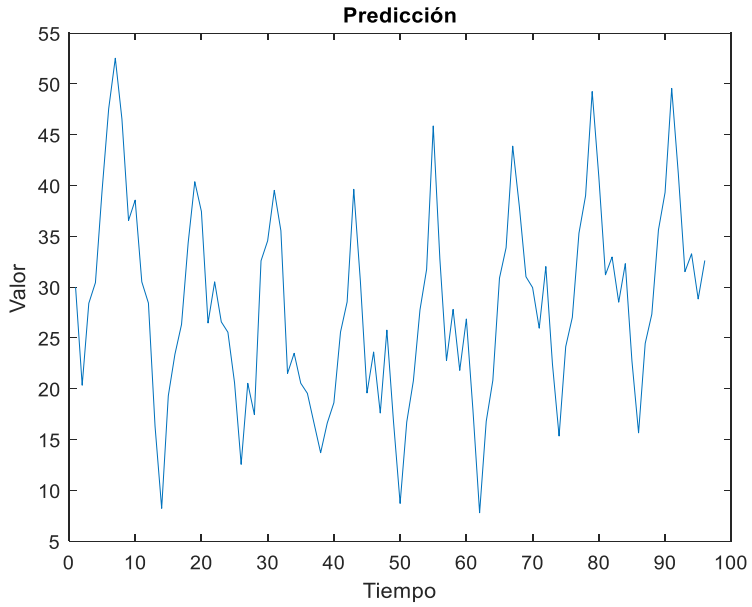

Figura 4-8: Predicción serie estacional

Como se ve se han predicho 24 puntos a partir de los 72 puntos de la serie inicial y los resultados obtenidos son muy buenos ya que sigue la misma tendencia que las 6 temporadas anteriores (cada temporada tiene una duración de temporada igual 12).

Una vez que ya hemos puesto en práctica el algoritmo de Holt-Winters al ejemplo anterior ya pasamos a tratar de aplicarlo a nuestra serie temporal de la segunda semana de Junio. Como he explicado anteriormente en el diseño la decisión fue pasarle al algoritmo una ventana completa (ventanas de 15 y 30 minutos) y tratar de predecir un período o una temporada y la ventana en concreto a la que se ha aplicado el algoritmo es la segunda ventana ya que en la fase de diseño llegamos a la conclusión de que en la segunda ventana se aprecia mejor la estacionalidad que en la primera. La segunda ventana de 15 minutos de la segunda semana de Junio es la siguiente:

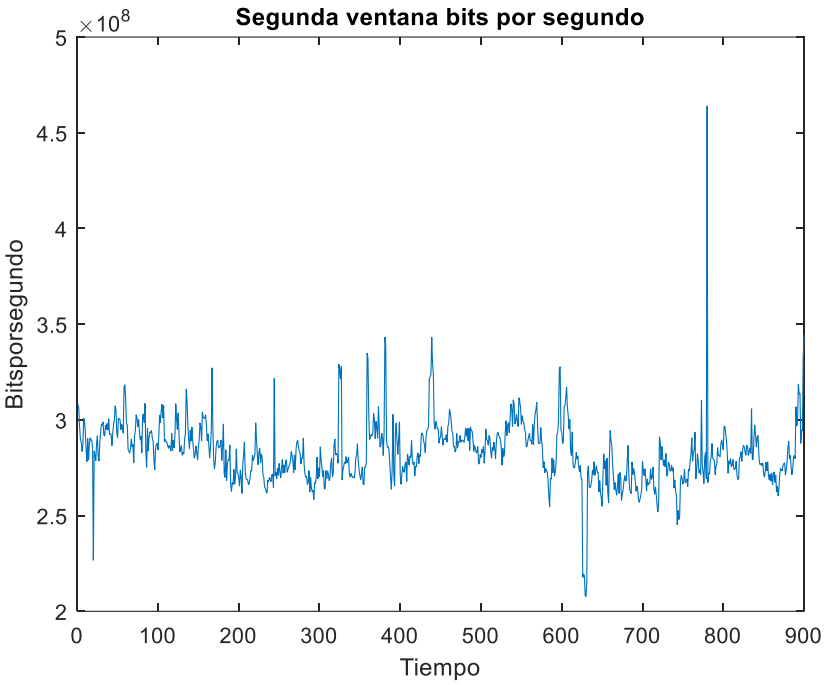

Figura 4-9: Segunda ventana 15 min bps

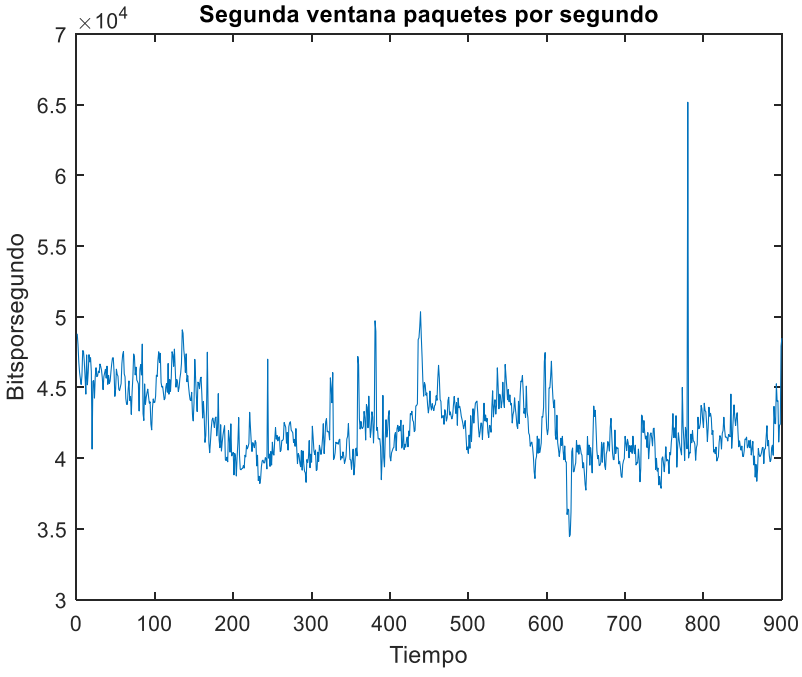

Figura 4-10 Segunda ventana 15 min paquetes por segundo

Para poder aplicar el algoritmo de Holt-Winters es imprescindible conocer la duración de la temporada, es decir, el período. En el ejemplo explicado anteriormente este período se percibía perfectamente, pero porque se trataba de un caso ideal. En este caso no se aprecia tan claramente, aunque en un principio mirando la gráfica se intuyó que estará en torno a 12,

pero no se sabía exactamente por lo que se procedió a calcular el período mediante dos algoritmos:

- Algoritmo Sf.
- Algoritmo Δf.

Estos dos algoritmos se implementaron en MATLAB. El algoritmo Sf [11] evalúa la posición de f (t_{n+i}) en relación con f(t_i) y f(t_{i+1}) y de lo contrario será evaluado como punto no barajado(NSP), siendo f la serie de la que se quiere calcular el período. El resultado será el máximo período la diferencia entre los puntos arrastrados y los no arrastrados.
El problema de estos dos algoritmos es que una de las variables que se tiene en cuenta para calcular el período de la serie es el período estimado por lo que estos algoritmos no son del todo exactos, entonces se utilizaron para obtener una idea de cuál sería el período o la duración de la temporada de la segunda ventana de la segunda semana de Junio y a partir de ahí comprobar cuál es la duración de la temporada que nos proporciona una mejor predicción. Suponiendo que el período de la segunda ventana de 15 minutos de la segunda semana de Junio es 12(es lo que vimos al ver la gráfica ampliada) obtenemos utilizando ambos algoritmos que el período o la duración de la temporada que tenemos que implementar en el algoritmo de Holt-Winters es 9.5.

Una vez ya conocidos la duración de la temporada tenemos que ver cuáles son los parámetros α, β, γ que nos van a dar una mejor predicción y éste ha sido uno de los principales problemas de este algoritmo ya que depende totalmente de la serie inicial (en este caso la segunda ventana de 15 minutos de la segunda semana de Junio).
A la hora de obtener los valores de estas variables matemáticas se utilizaron dos metodologías:
La primera la denominamos fuerza bruta ya que lo que hacíamos para obtener los valores de estas variables matemáticas es hacer tres bucles for() que vayan recorriendo α, β, γ desde 0 hasta 1, tomando 100 valores entre 0 y 1 y representar la predicción que nos de menor error con respecto a lo que se esperaría. Para ello se tuvo en cuenta el concepto de la distancia entre dos rectas explicadas en el estado del arte.
La segunda metodología y la que finalmente utilizamos fue utilizar simulated annealing con gradiente que es un método probabilístico para aproximar el óptimo global de una función dada. Dentro del algoritmo hay un término que es la precisión, pero en los resultados no afecta demasiado utilizar un valor u otro. La ventaja de este método frente a la fuerza bruta es la velocidad a la hora de obtener resultados ya que en este método no se comprueban todos los valores posibles de las variables matemáticas α, β, γ como se hacía con el método de la fuerza bruta.

En primer lugar, se ha tratado de predecir el primer período o temporada de la tercera ventana de Junio. Como no sabíamos cuál era la duración de la temporada de la serie inicial (segunda ventana completa de Junio) lo que se ha hecho es ir probando para diferentes períodos (teniendo en cuenta que el período obtenido anteriormente es 9.5) y se ha llegado a la conclusión de que la duración de la temporada o el período real de la serie es 11. El primer período de la tercera temporada en bits por segundo debería ser así:

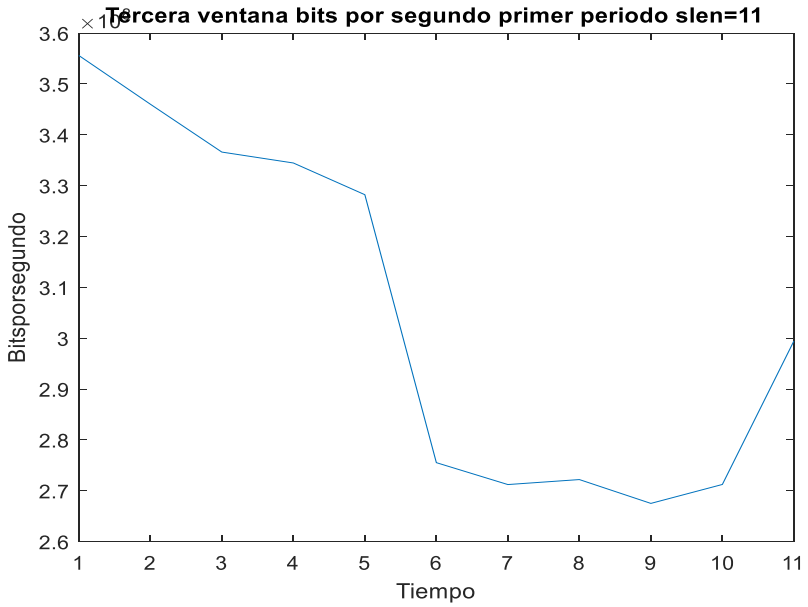

Figura 4-11: Primer período tercera ventana bps con período 11

El resultado obtenido mediante el algoritmo de Holt Winters es el siguiente:

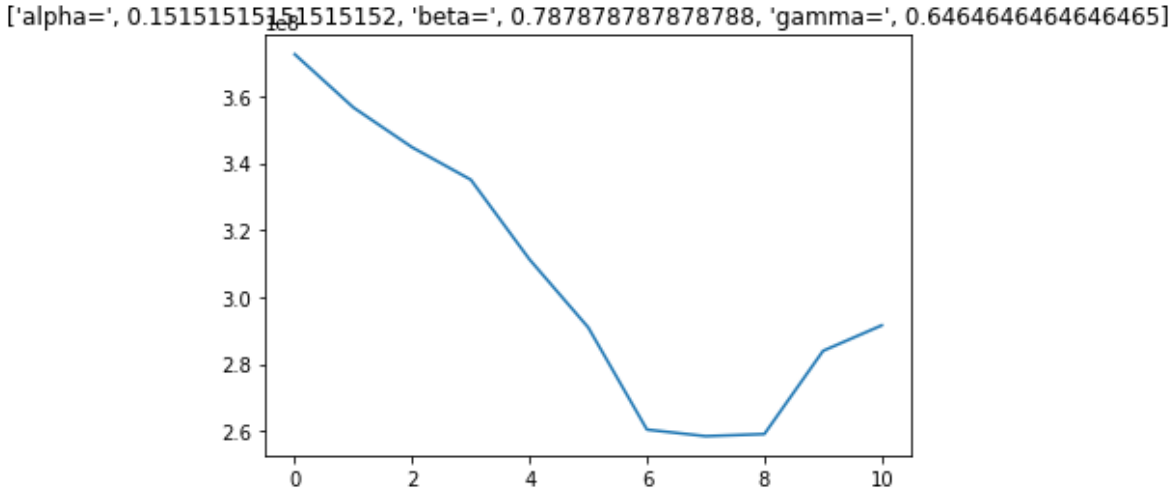

Figura 4-12: Predicción primer período tercera ventana bps con período 11

Como vemos la predicción obtenida mediante el algoritmo de Holt Winters es muy buena ya que sigue la misma tendencia que se esperaba. En la primera gráfica en el título hay una variable slen=11, eso lo que quiere decir es el período. En todo el desarrollo se ha predicho el mismo número de puntos que el período de la serie.

A continuación, se ha realizado el mismo procedimiento para los paquetes por segundo. El primer período de la tercera temporada en paquetes por segundo debería ser así:

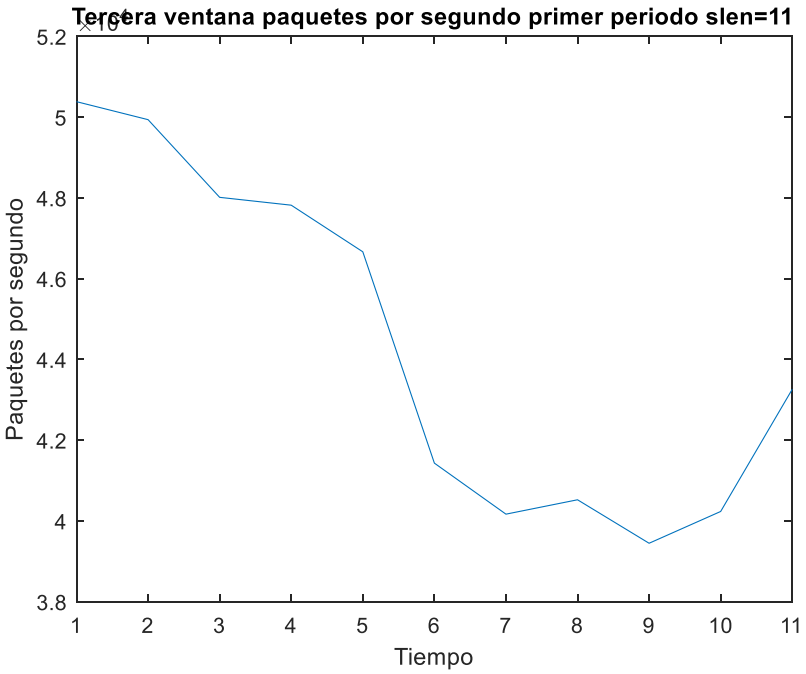

Figura 4-13: Primer período tercera ventana paquetes por segundo

El resultado obtenido mediante el algoritmo de Holt-Winters es el siguiente:

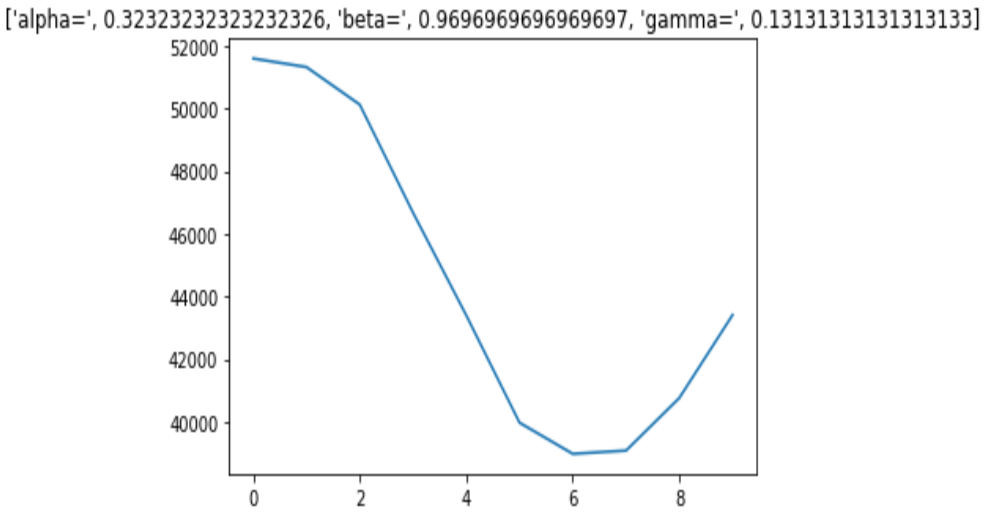

Figura 4-14: Predicción primer período tercera ventana paquetes por segundo

Como vemos la predicción obtenida mediante el algoritmo de Holt-Winters para los paquetes por segundo es buena, pero no tanto como para los bits por segundo, pero igualmente nos sirve para comprobar que el algoritmo funciona de manera adecuada. A partir de este momento únicamente vamos a trabajar con los bits por segundo ya que los resultados en cuanto a calidad en la predicción van a ser muy similares.

Una vez realizada la predicción del primer período de la tercera ventana con ventanas de 15 minutos y con período 11 se procedió a calcular más períodos para ver si la predicción era buena justo en ese período o era realmente buena ya que esa duración de la ventana y ese período eran los correctos para la serie de partida. Anteriormente habíamos utilizado ventanas de 15 minutos y un período igual a 11, pero como he explicado anteriormente al calcular el período en nuestra serie con los dos algoritmos anteriores (Algoritmo Sf y algoritmo Δf) el resultado del período fue 9.5 por lo que decidimos calcular la predicción de más períodos para ventanas de 15 minutos y períodos 10 y 11. Además pensamos que con ventanas de 30 minutos los resultados de la predicción podrían ser más precisos al tener más datos, así que también se calcularon las predicciones para ventanas de 30 minutos y períodos 10 y 11. Los resultados obtenidos fueron los siguientes:

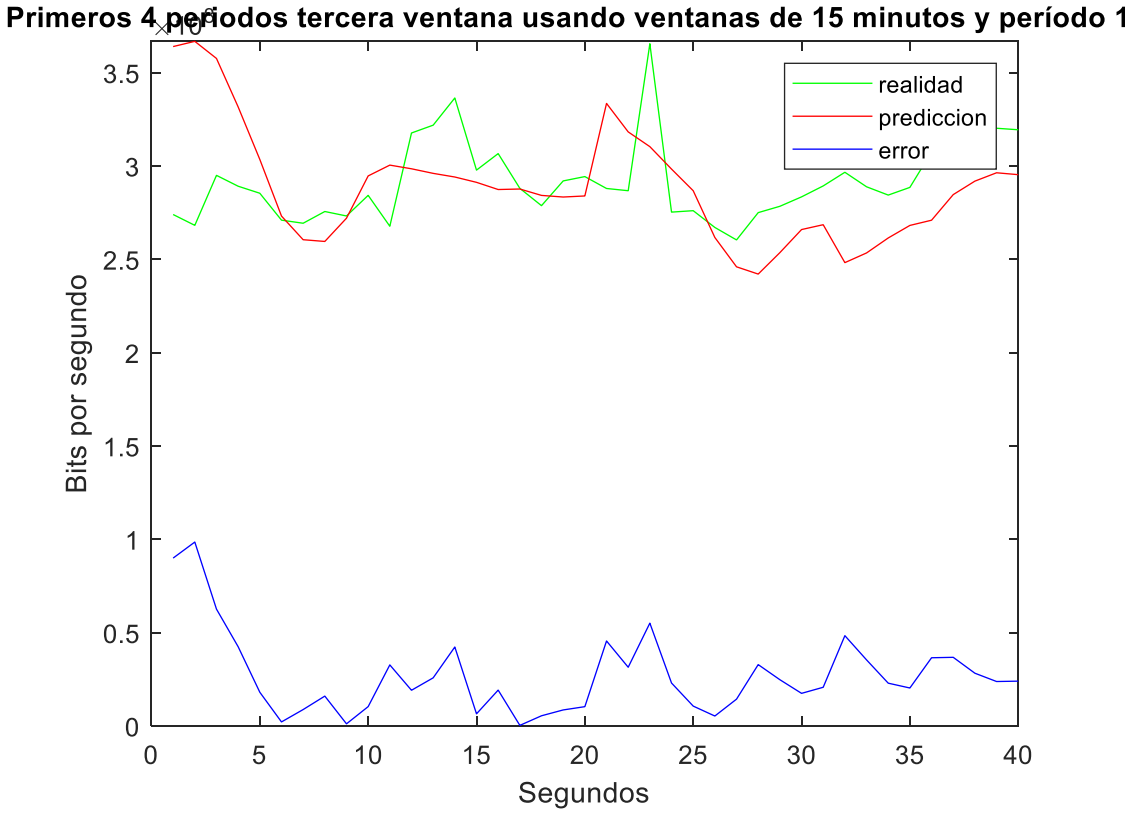

Figura 4-15: Predicción 4 primeros períodos usando ventanas de 15 minutos y período 10

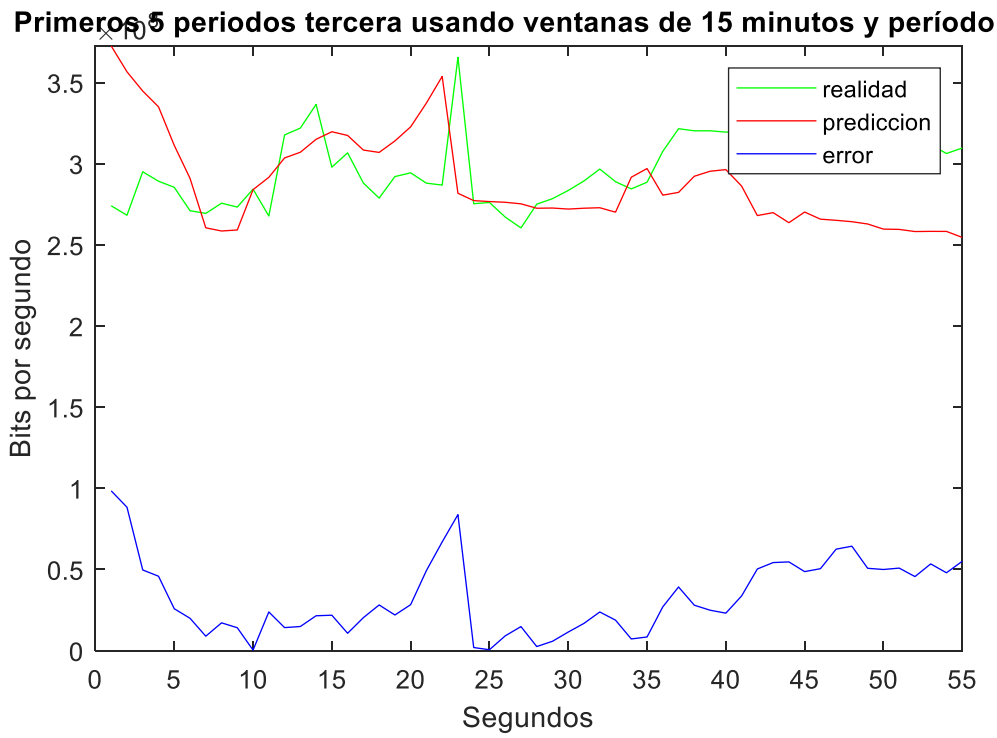

Figura 4-16: Predicción 5 primeros períodos usando ventanas de 15 minutos y período 11

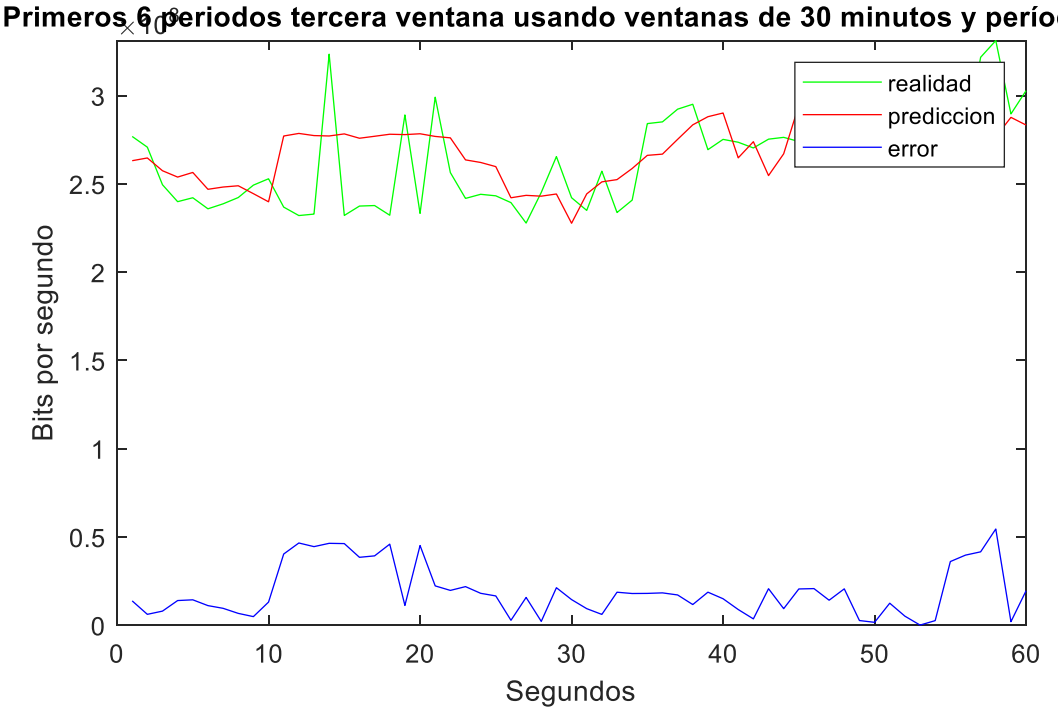

Figura 4-17: Predicción 6 primeros períodos usando ventanas de 15 minutos y período 10

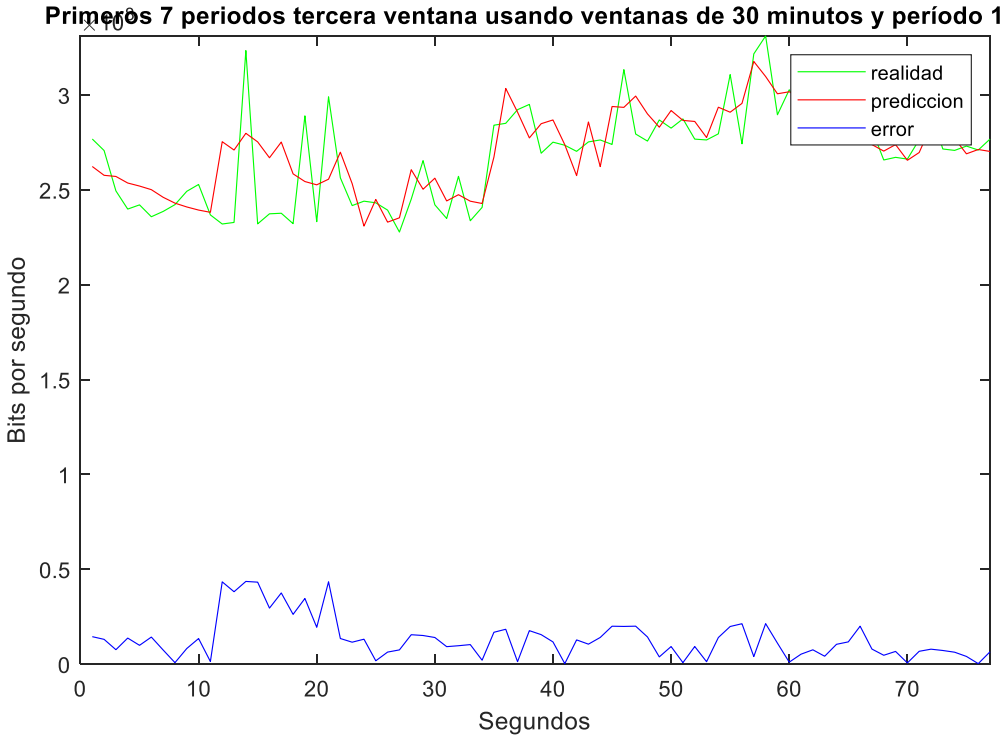

Figura 4-18: Predicción 7 primeros períodos usando ventanas de 30 minutos y período 11

Una vez obtenidas las cuatro predicciones había que escoger una de las cuatro posibilidades ya que el objetivo final era poder predecir 10 y 15 minutos, introducir un ataque y ver como se comportaba frente a un ataque este algoritmo de Holt-Winters. La decisión final fue utilizar una ventana de 30 minutos para así tener en cuenta más datos en la predicción y un período igual a 11 ya que al comparar la predicción realizada para una ventana de 30 minutos y período 10 y la predicción realizada para una ventana de 30 minutos y período 11 hay un poco menos de error para período 11, aunque seguramente los resultados hubieran sido muy similares.
A partir de ahí se calculó la predicción de los 10 primeros minutos de la tercera ventana a partir de la segunda ventana de duración igual a 30 minutos y de período 11 y se obtuvieron los siguientes resultados:

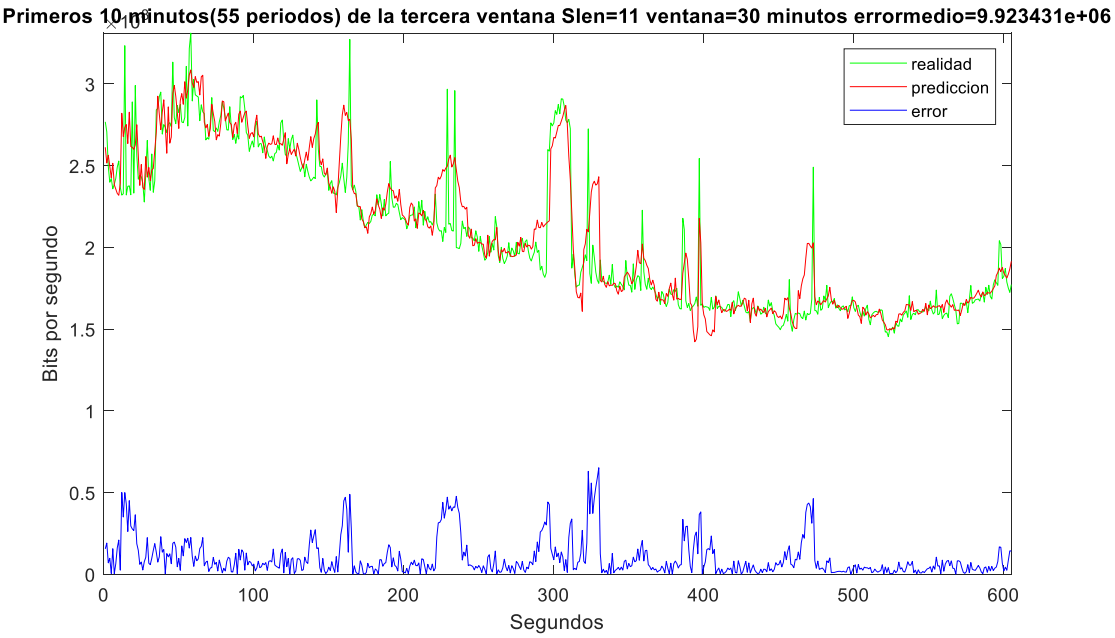

Figura 4-19 Predicción 10 primeros minutos tercera ventana de 30 minutos

Una vez visto que la predicción de los 10 primeros minutos de la tercera ventana es muy buena, ya que el error medio es muy bajo en comparación al tráfico en bits por segundo se pasó a introducir un ataque para ver si podíamos detectarlo con el algoritmo de Holt-Winters. El ataque que introdujimos era un ataque que tenía una duración de 10 minutos. Por lo tanto, realizamos una predicción de 15 minutos para así poder ver como el ataque nos produce un error en la predicción mientras está presente, pero una vez que se finaliza la predicción vuelve a ser muy buena. A continuación, se muestra la predicción de los 15 primeros minutos de la tercera ventana de 30 minutos de la segunda semana de Junio:

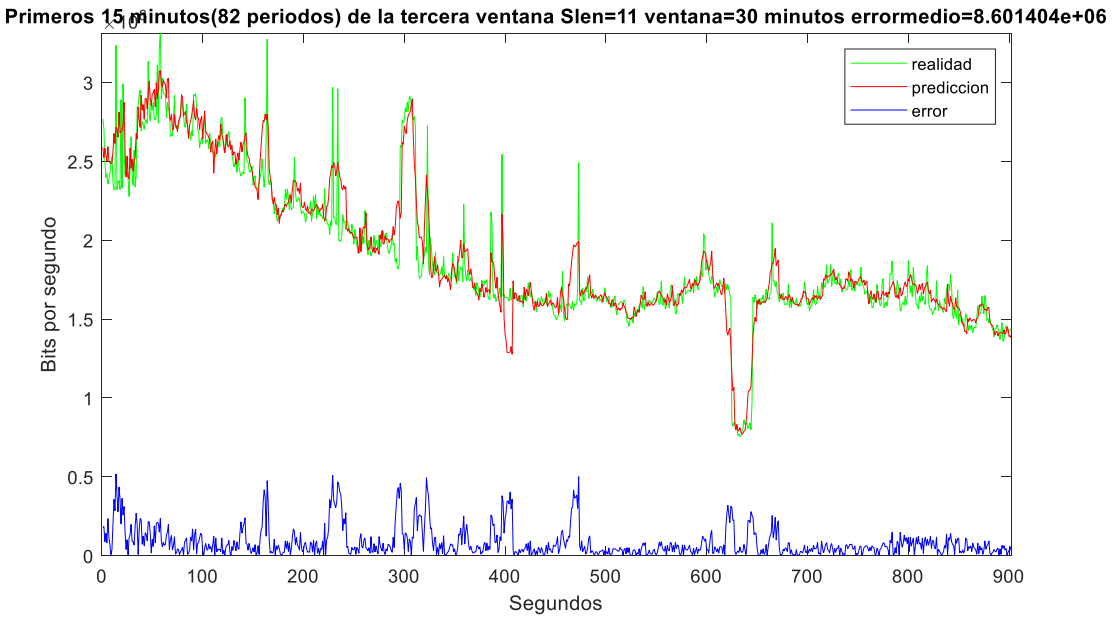

Figura 4-20 Predicción 15 primeros minutos tercera ventana de 30 minutos

Como se puede ver los resultados obtenidos son muy buenos, ya que el error medio cometido es muy bajo en comparación al tráfico real en bits por segundo e incluso hemos conseguido tener un error un poco menor que cuando realizábamos la predicción de los 10 primeros minutos.
Una vez que ya hemos obtenido la predicción el objetivo era comprobar que el algoritmo de Holt-Winters nos permitía detectar ataques. A continuación, hemos representado los ataques que se producen durante la segunda semana de Junio(datos que estamos analizando)[8]

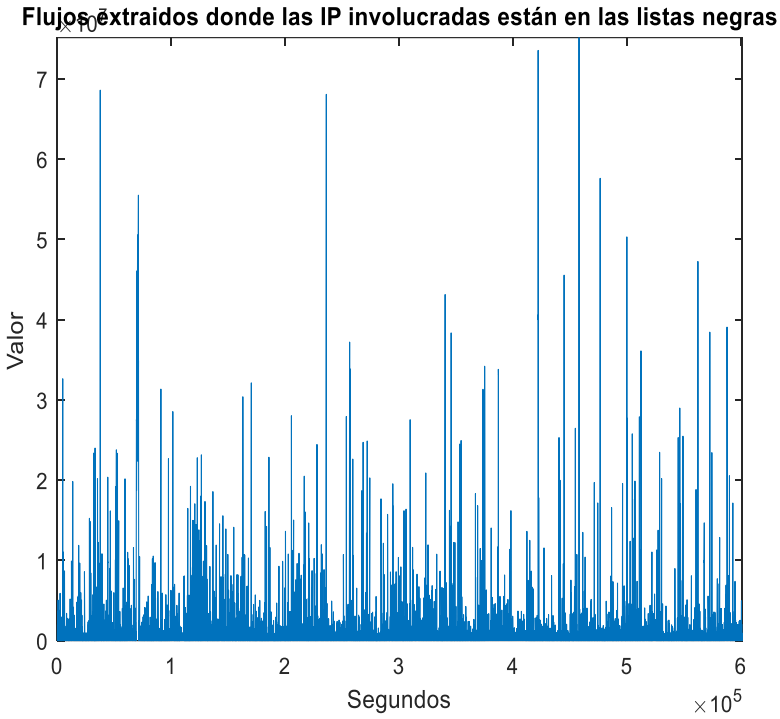

Figura 4-21 Ataques donde las IP están en listas negras

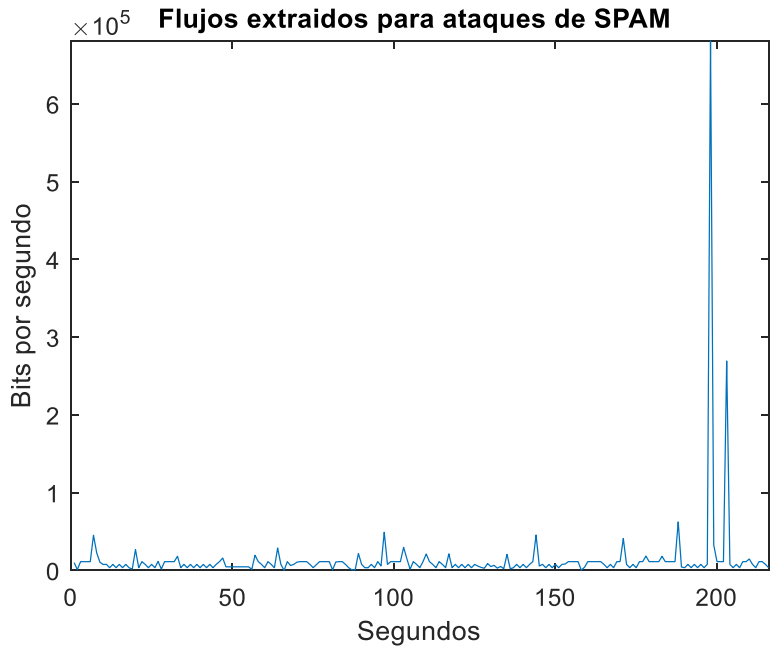

Figura 4-22 Ataques de spam

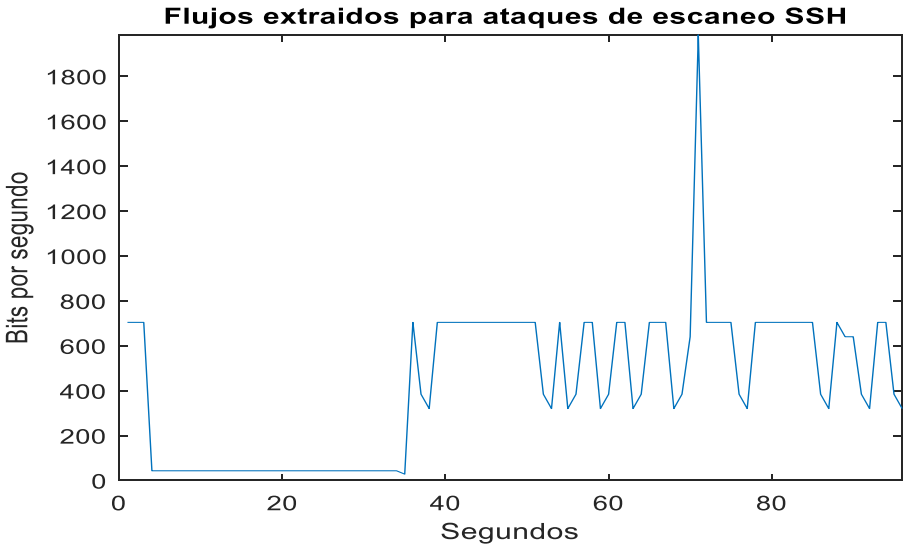

Figura 4-23 Ataques de escaneo SSH

Como vemos al comparar los órdenes de magnitud del tráfico en bits por segundo de los tres ataques en con el tráfico en bits por segundo de los 15 primeros minutos de la tercera ventana vemos que los ataques van a ser insignificante y por tanto es muy probable que no se puedan detectar los ataques. A continuación, he representado una gráfica en la que se muestra como afectarían esos tres ataques sumados en nuestra predicción:

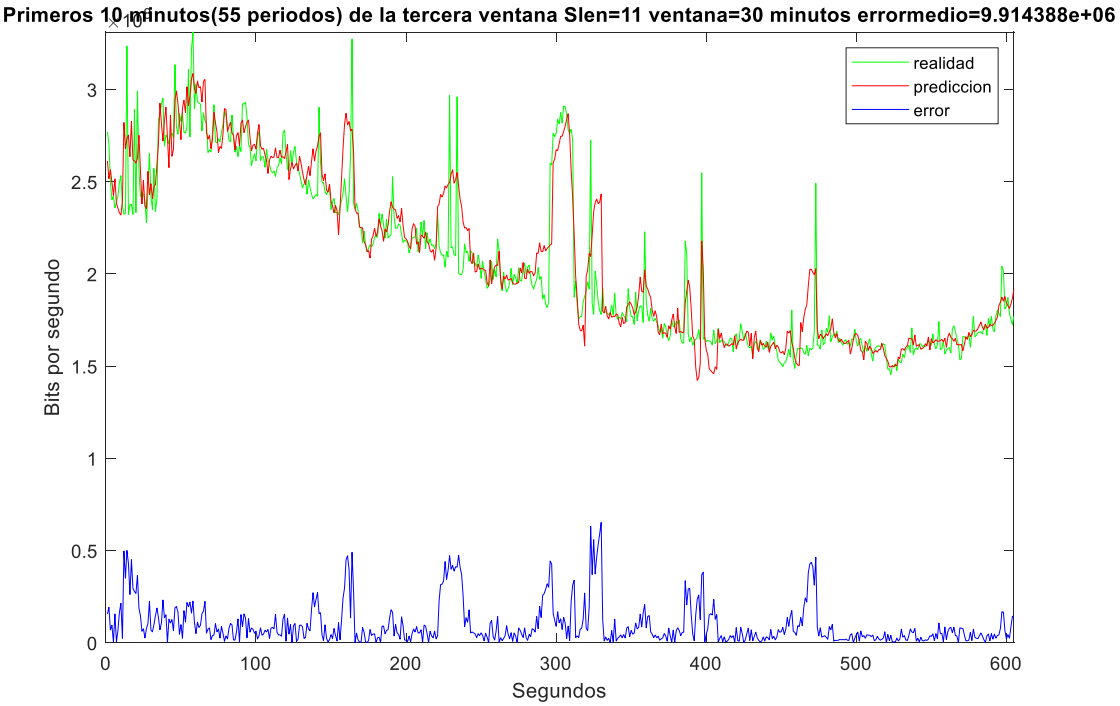

Figura 4-24: Error en la detección ataques segunda semana de Junio

Como vemos el error (la diferencia entre nuestra predicción y nuestro tráfico normal o real con el ataque incluido) cometido en la predicción con el ataque incluido no ha aumentado con respecto al error que teníamos con el tráfico normal, incluso en este caso baja. Esto puede ocurrir porque al sumar el ataque a nuestro tráfico normal ha dado la casualidad de que nuestra predicción había sido un poco más alta que nuestra realidad en cuanto a tráfico en bits por segundo y por tanto la diferencia entre nuestro tráfico normal más el ataque y nuestra predicción va a ser un poco inferior.

Viendo que los ataques anteriores no se podían detectar se procedió a analizar otros ataques [12], en este caso hemos tratado de detectar los ataques de denegación de servicio existentes y nuestros resultados ahora son los siguientes:

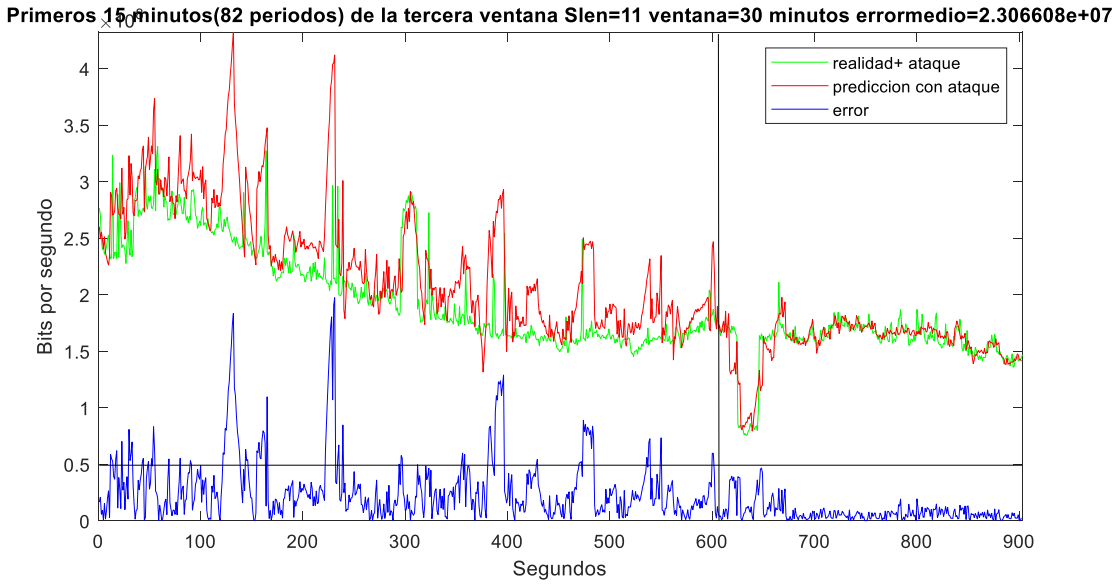

Figura 4-25: Detección ataque DoS grande

Analizando los resultados vemos que el error cometido es mucho mayor que cuando el ataque finaliza (a partir del segundo 605). En la gráfica hemos puesto una línea horizontal y una vertical con las que apreciamos que el error cometido cuando se produce el ataque supera $0.5*10^8$ bits por segundo (que era el error medio que se producía cuando no había ningún tipo de ataque), mientras que cuando finaliza el ataque el error es menor. Por lo tanto, vemos el ataque DoS que se produce durante los primeros 605 segundos (aproximadamente 10 minutos) se detecta perfectamente.
A continuación, se introdujo un ataque DoS de un menor tráfico en bits por segundo obteniendo los siguientes resultados:

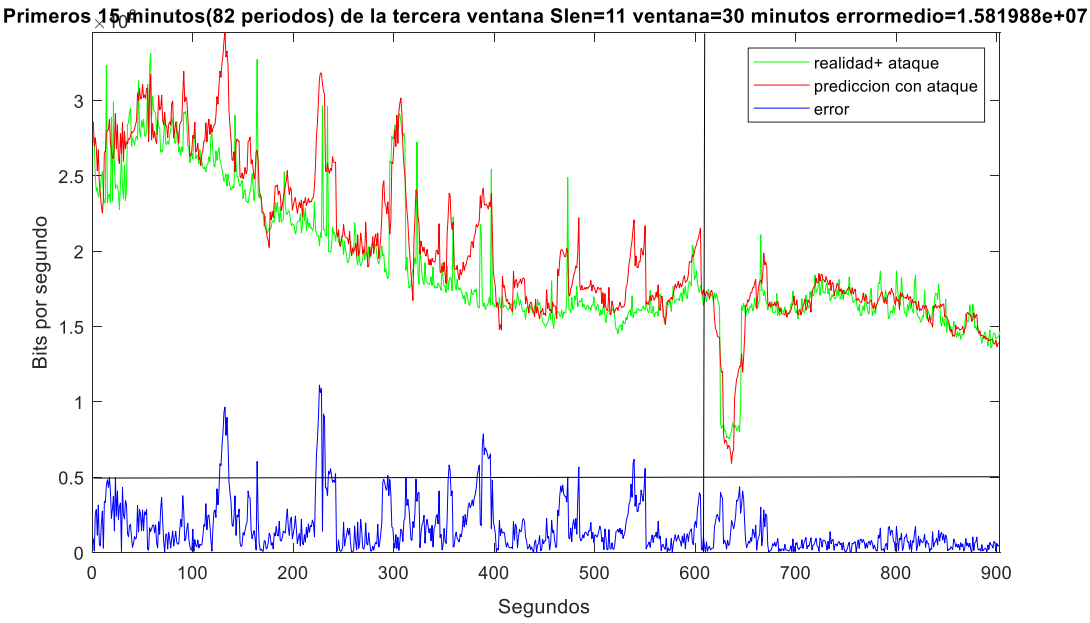

Figura 4-26 Detección ataque DoS pequeño

Analizando la gráfica vemos que los errores que se producen durante el ataque superan $0.5*10^8$ bits por segundo (que era el error medio que se producía cuando no había ningún tipo de ataque), mientras que cuando finaliza el ataque este el error es menor. Ahora vemos que al introducir un ataque de menor tráfico en cuanto a bits por segundo el error que se produce entre la predicción y la zona de tráfico normal afectada por el ataque es menor.

4.4 Conclusiones

A lo largo de este capítulo hemos ido observando, analizando y explicando todos los procesos que se han desarrollado a lo largo de este trabajo. En primer lugar, se analizaron los aspectos referentes al suavizado exponencial simple y doble para posteriormente aplicar esos conceptos para poder desarrollar el algoritmo de Holt-Winters. Posteriormente se analizaron los resultados y datos usados por los trabajos anteriores llegando a la conclusión de que había que utilizar otros datos. A continuación, se analizaron características imprescindibles para poder aplicar el algoritmo de Holt-Winters a nuestra serie real y así obtener una predicción
del tráfico correcta. Finalmente se introdujo un ataque y se comprobó que se podía detectar correctamente.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

En este trabajo partíamos de tres trabajos anteriores en los que mediante el uso de los parámetros alfaestables se diferenciaba el tráfico normal, el tráfico de ataque y el tráfico mixto. El objetivo de este trabajo era hacer un análisis del tráfico de Internet mediante el uso del suavizado exponencial y a partir de ahí tratar de aplicar el algoritmo de Holt-Winters o también conocido como suavizado exponencial triple para poder predecir el tráfico de Internet.

En primer lugar, se desarrollaron algunos métodos de pronóstico sencillos y el suavizado exponencial simple y doble para conocer algunas de las variables y sus diferencias con respecto al algoritmo de Holt-Winters. Una vez ya conocidas algunas de las variables del suavizado simple y doble se pasó a desarrollar el algoritmo de Holt-Winters.

En un principio se aplicó el algoritmo a una serie "ideal" (ya que la duración de la temporada se veía perfectamente) se desarrolló el algoritmo ya conociendo las variables matemáticas (α, β, γ) y se llegó a la conclusión de que el algoritmo funcionaba correctamente.

A partir de ahí la idea era desarrollar este algoritmo a los datos proporcionados por los trabajos anteriores, pero estos datos no eran estacionales, lo cual es punto obligatorio para poder aplicar el algoritmo de Holt-Winters, esto nos obligó a utilizar otros datos distintos, en este caso los referentes a la segunda semana de Junio. Una vez escogidos los datos de la segunda semana de Junio que estaban en un formato .csv (fichero de texto separado por comas) se desarrolló un algoritmo AWK para quedarnos únicamente con las características que nos interesaban (tiempo en segundos, bits por segundo y paquetes por segundo).

En este momento fue en el que nos empezamos a hacer preguntas importantes a la hora de hacer nuestro estudio: ¿Cuál es la duración de la temporada?, ¿Qué tamaño de la ventana vamos a usar?, ¿Cuáles son los valores de α, β, γ que nos van a proporcionar una mejor predicción? Para conocer la duración de la temporada o período desarrollamos dos algoritmos explicados anteriormente, pero estos algoritmos no eran demasiado precisos, lo cual provocó que tuviéramos que ir probando diferentes períodos. En cuanto al tamaño de la ventana se utilizaron ventanas de 15 y 30 minutos ya que nos daban los mejores resultados posibles y los valores de las variables matemáticas α, β, γ se calcularon de manera que nuestro error entre nuestra predicción y el tráfico real fuera el menor posible. Una vez resueltas estas cuestiones vimos que nuestros resultados en las predicciones fueron muy buenos tanto cuando se decidió predecir períodos pequeños con ventanas de 15 y 30 minutos como cuando se realizó la predicción de los 10 y 15 primeros minutos de la tercera ventana a partir de la segunda ventana, utilizando ventanas de 30 minutos.

Una vez obtenida una buena predicción se trató de introducir ataques de diferentes tipos con la idea de predecir esos ataques, para ello se fue mirando como eran los diferentes ataques existentes que nos podían afectar, se escogieron diferentes ataques y se comprobó que con el algoritmo de Holt-Winters podíamos detectar dichos ataques. Por lo tanto, hemos visto que el algoritmo de Holt-Winters es un método muy útil y con buenos resultados a la hora de predecir el tráfico y además nos permite detectar ataques en caso de que éstos se produzcan.

5.2 Trabajo futuro

De cara a un futuro se podría tener en cuenta aspectos como:

- Aplicar el algoritmo a otras semanas distintas
- Tratar de detectar nuevos ataques
- Desarrollar un nuevo algoritmo para conocer el período

Referencias

[1] Q. T. Tran, L. Hao, and Q. K. Trinh, "CELLULAR NETWORK TRAFFIC PREDICTION USING EXPONENTIAL SMOOTHING METHODS," 2019. Accessed: Jun. 17, 2021. [Online]. Available: http://www.jict.uum.edu.my/images/vol18no1jan19/1-18.pdf.

[2] "Seguridad en Internet: Los ataques de red alcanzan su nivel más alto en los últimos dos años." https://www.redseguridad.com/actualidad/los-ataques-de-red-alcanzan-su-nivel-mas-alto-en-los-ultimos-dos-anos_20201221.html (accessed May 12, 2021).

[3] "Suavización exponencial - Qué es, definición y concepto | 2021 | Economipedia." https://economipedia.com/definiciones/suavizacion-exponencial.html (accessed Jun. 17, 2021).

[4] "Distancia - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Distancia (accessed Jun. 17, 2021).

[5] "Ciberataque - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Ciberataque (accessed Jun. 17, 2021).

[6] "¿Qué son los ataques DoS y DDoS? | Oficina de Seguridad del Internauta," Aug. 21, 2018. https://www.osi.es/es/actualidad/blog/2018/08/21/que-son-los-ataques-dos-y-ddos (accessed Jun. 17, 2021).

[7] E. Revuelta Santiago, "Trabajo Fin de Grado," Universidad Autónoma Madrid, 2020.

[8] "UGR'16 Dataset." https://nesg.ugr.es/nesg-ugr16/june_week2.php#INI (accessed Jun. 17, 2021).

[9] "Holt-Winters Forecasting for Dummies (or Developers) - Part I - Gregory Trubetskoy," *29-01-16*. https://grisha.org/blog/2016/01/29/triple-exponential-smoothing-forecasting/ (accessed May 06, 2021).

[10] "Holt-Winters Forecasting for Dummies - Part II - Gregory Trubetskoy." https://grisha.org/blog/2016/02/16/triple-exponential-smoothing-forecasting-part-ii/ (accessed Junio. 17, 2021).

[11] J. D. A. Rairán, "Two algorithms for estimating the period of a discrete signal," *Ing. e Investig.*, vol. 34, no. 3, pp. 57–63, 2014, doi: 10.15446/ing.investig.v34n3.41943 Accesed: Marzo. 2 2021.

[12] "UGR'16 Dataset." https://nesg.ugr.es/nesg-ugr16/august_week2.php#INI (accessed Jun. 17, 2021).

Glosario

API	Application Programming Interface
DoS	Denial of Service
DDoS	Distributed Denial of Service
ARIMA	Media móvil integrada autorregresiva

Anexos

A Manual de instalación

B Manual del programador

Todo el código desarrollado está subido al siguiente repositorio

https://github.com/eduardocabornero/TFG

C Anexo …